Maastricht University

Study on the effect on the development of the information society of European public bodies making their own software available as open source

Citation for published version (APA):

Ghosh, R. A., Glott, R., Gerloff, K., Schmitz, P-E., Aisola, K., & Boujraf, A. (2007). *Study on the effect on the development of the information society of European public bodies making their own software available as open source*. European Commission.

Document status and date:
Published: 01/01/2007

Document Version:
Publisher's PDF, also known as Version of record

Please check the document version of this publication:

• A submitted manuscript is the version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal.

If the publication is distributed under the terms of Article 25fa of the Dutch Copyright Act, indicated by the “Taverne” license above, please follow below link for the End User Agreement:

www.umlib.nl/taverne-license

Take down policy
If you believe that this document breaches copyright please contact us at:

repository@maastrichtuniversity.nl

providing details and we will investigate your claim.

Download date: 06 May. 2023

Study on the effect on the development of the information society of European public bodies making their own software available as open source

D8: Final report

Version 2.0

Authors: Rishab A. Ghosh (MERIT)
Rüdiger Glott (MERIT)
Karsten Gerloff (MERIT)
Patrice-Emmanuel Schmitz (UNISYS)
Kamini Aisola (UNISYS)
Abdelkrim Boujraf (UNISYS)

The opinions expressed in this Study are those of the authors and do not necessarily reflect the views of the European Commission.

April 19, 2007

Copyright
This report is copyright © 2007 the European Commission.

Table of contents

Executive Summary

The opinions expressed in this Study are those of the authors and do not necessarily reflect the views of the European Commission.

Publishing software fully owned by public bodies as Free/Libre/Open Source Software (FLOSS)[1] could facilitate re-use, adaptation and modification of the software by other public organisations, as well as other actors. The free availability of public sector software could possibly result in a "multiplier effect", which if significant, could have an accelerating effect on the development and take- up of information society technologies.

The FLOSS paradigm could be a practical and operational way of allowing the above described multiplier effect to take place. FLOSS has leapt to prominence by taking significant share in some specific segments of the software infrastructure market.

This report addresses the following points and questions:

- What would be the potential impact on the development of the Information Society if public organisations (administrations, research institutions, universities, agencies, public companies etc.) were to release software fully owned by them under a FLOSS licence? Is an amplification effect on the adoption and use of information society technologies to be expected? What are the possible characteristics of such an effect, and under what circumstances would it occur?
- What are the conditions under which software fully owned by public organisations can be made available using a FLOSS licence? What are the opportunities, barriers and limitations to this process?
- What options and recommendations can be provided for actions that could be taken, notably by the European Commission, EU Member States, and the private and public sector in general?

The study draws upon data gathered in the 2004 FLOSSPOLS survey of 955 public authorities in 13 EU member states. To examine certain aspects in more detail, we also studied six cases of FLOSS development and distribution by public authorities in four different European countries. Possible impacts on the European software market were assessed with the help of theoretical structures based on the standard models of industrial organisation by Cournot (1838) and Hotelling (1929).

Opportunities and barriers

Challenges and barriers for public bodies that are looking to use FLOSS and distribute their own software as FLOSS are:

- Public administrations wanting to develop, distribute or adapt FLOSS will need to build up a certain degree of technical capabilities, whether by employing skilled staff or by contracting external services. However, lack of support for the *use* of FLOSS does not appear to be an important problem.

[1] In this report we refer to the single phenomenon known by the various terms "libre software", "free software" and "open source software" as Free/Libre/Open Source Software (or FLOSS). We note that the EU/FP5 FLOSS developer survey of over 2800 respondents showed that a majority of developers themselves identify with the term "free software", while Libre software (logiciel libre, software libre, software libero) is the favoured term in southern Europe and Latin America.

- With over 100 FLOSS licences in existence, special expertise is sometimes required to determine how a given piece of source code can be legally distributed. This is especially true when combining software from pieces distributed under different licences.
- To develop software as FLOSS, a public administration's staff needs to master the appropriate skills. Besides programming knowledge, this includes both the management of a development project and its code base, and communication with other developers and the wider FLOSS community. A number of procedures have to be developed and put in place, e.g. for integrating feedback.
- The challenges associated with developing and distributing FLOSS require a redistribution of tasks within the organisation's IT department. People who were previously tasked with maintenance or performing internal processes become responsible for communication with actors outside the PA, for providing content for websites and mailing lists, and for adapting processes to new technologies and needs.
- The overwhelming majority of communication about FLOSS development happens in English. This might constitute something of a language barrier for some public administrations.
- Lack of prior experience with FLOSS is likely to be a barrier to using and releasing FLOSS software.

Opportunities are:

- The formation of a pool of cost-efficient public sector software of potentially high quality.
- Possibly significant cost savings through the reuse of software made available by other public bodies. Most cases studied had the perception that their costs had decreased through using FLOSS. But such savings may not be the top priority, especially when the decision to develop and release FLOSS is made with strategic objectives in mind. The externalisation of development costs plays a comparatively minor role.
- Increased flexibility in software use, both through independence from vendors and from a deeper understanding of how the software works.
- Since dealing with FLOSS induces public administrations to deal with questions of copyright, patents and trademarks, legal knowledge is often generated in institutions where it was not present before.
- increased interoperability, both within the administration's IT system and with citizens and enterprises, as FLOSS uses open standards and protocols. Exchanging data with other institutions becomes easier.

Impact

The study examined the impact the development and distribution of FLOSS by public bodies has on eGovernment services, the economy, and the information society.

- The degree to which an administration's software configuration and hardware architecture are affected by FLOSS development and distribution projects in the public sector depends on the scope and purpose of these projects. If the project aims at reorganising the services the public body provides, then the effect on software configuration and hardware architecture is quite high. In contrast, when FLOSS is produced for the purpose of serving the needs of other users / institutions or as an application that runs by and large

independently from the services and processes of other departments, the impact on software configuration and hardware architecture is rather low.

- As for the impact of FLOSS releases on the economy, neither the "releasers" nor the "non-releasers" expect a negative economic impact of public bodies releasing their own software as FLOSS. Quite to the contrary, there is some confidence that it might have some positive economic impact, if any.
- However, our analysis of the case studies shows that local businesses and the local FLOSS community benefit from the general need of public administrations to adapt software to their specific needs, and these spill-over effects – though varying from case to case - are very important at the political level of each project.
- The cases that were analysed showed strong differences with regard to their impact on the information society. The greatest impact clearly occurs when the decision to release software as FLOSS is not made for purely technical or practical reasons, but is part of a strategy aiming at a wider impact on the information society from the start, as in Extremadura. On the other hand, the impact of merely releasing a specialised application as FLOSS is likely to be limited.
- Though we consider that *currently* the *actual* impact of such attempts as limited, we see a great *potential* for a fundamental impact of public bodies' FLOSS activities *in future*. Based on the analysis of the case studies, we distinguish four different scenarios for FLOSS development by public bodies, each with their own potential impact on the information society.

Conclusions

Based on the analysis of our observations, the central conclusions of the study are:

- The type of software predominantly developed by or for public bodies is neither general-purpose nor public sector specific software, but specialised software such as content management or work-flow systems, heavily customised to be of use to the public sector but possibly also useful to other organisations.
- Legal issues did not prove to be a major barrier for public bodies to engage in FLOSS development projects. The present legal environment appears sufficient to enable public bodies to release their software as FLOSS.
- Instead, we identified as the most important barrier a lack of IT professionals that are experienced in software development and / or FLOSS. The availability of skills is the bottleneck for the public sector to become capable of developing and distributing FLOSS.
- While FLOSS-related skills – or rather the lack thereof – frequently are a barrier to developing FLOSS, those administrations who engaged in development reported an improvement of their skill base.
- Except for one case, all FLOSS projects were initiated by the IT department of the public body involved. Managers and employees in public administrations still tend to focus on their organisation and to disregard the potential of direct interaction with citizens and users and the FLOSS community.
- The use and development of FLOSS in public bodies should not be discouraged; instead, it is important to build awareness for the advantages that this software model may bring. Governments (on all levels, i.e. local, regional, national, EU) should make decision-makers

in public administrations aware of the advantages of developing and distributing their own software under a FLOSS licence.

- The impact on the information society of FLOSS releases by public bodies depends on the question whether such an impact is intended. If it occurs, it usually is the result of a conscious effort.

In the Annex to this report, we provide guidelines and a practical toolkit for making software owned by public bodies available under an appropriate FLOSS licence.

1 Introduction

The Lisbon European Council of March 2000 set the objective of making Europe the most competitive and dynamic knowledge-based economy in the world within 10 years. The goals of the Lisbon agenda include the provision of a favourable environment for investment, the modernization of public services, the creation of jobs, boosting productivity, and giving everyone the opportunity to participate in the Information Society. Dissemination of best practice and ensuring greater convergence between EU Member States were considered important

An important element in reaching the above goals is the stimulation of services and applications. Community actions for promoting the adoption of Information and Communication Technologies (ICTs) in administrations, in public sector services and in businesses have mostly been carried out under the eEurope initiative[2]. These include the exchange of experiences, of good practices and demonstration projects, but also the sharing of lessons from failures. In particular projects were to be launched to accelerate the roll-out of leading edge applications and infrastructure.

A background research paper funded under the IDABC program dealing with pan-European eGovernment Services (http://europa.eu.int/idabc/en/home) underlines the positive impact that eGovernment could carry on competitiveness and growth ("The impact of e-Government on competitiveness, growth and jobs", February 2005, available from IDABC). In recent times a debate has emerged on the issue of software fully developed with taxpayers' money and fully owned by public organisations. One of the main considerations in this debate is whether such software could be made available as Free/Libre/Open Source Software (FLOSS)[3], giving equal and non-discriminatory access to everyone for further use and/or modification and/or re-distribution. As software is a non-rivalrous resource not suffering from use-induced depletion it could be made available on an access-to-all basis. Not only is software a non- rivalrous good, but actually its usefulness could increase from wide-spread use due to the well-known networking effect.

To the extent that business processes have sufficient similarity, publishing software fully owned by public bodies could facilitate re-use, adaptation and modification of the software by other public organisations, including administrations, agencies, research institutions, public companies, etc. More importantly, the availability of software owned by or fully paid for by public organisations could generate new business opportunities and increase the knowledge available in the public sphere.

The mechanism of re-using, changing and enhancing software developed by other public bodies could possibly result in a "multiplier effect", which if significant, could have an accelerating effect on the development and take- up of information society technologies. It might reduce duplication of development efforts and could contribute to the quality and economic use of software in the public domain.

The FLOSS paradigm could be a practical and operational way of allowing the above described multiplier effect to take place. FLOSS is software distributed according to license terms which allow, inter alia, for access to the source code of the software, the right to use the software and to develop derived products, provided that derivative works are distributed under such license terms or

[2] Decision 2256/2003/EC of the Parliament and of the Council

[3] In this report we refer to the single phenomenon known by the various terms "libre software", "free software" and "open source software" as Free/Libre/Open Source Software (or FLOSS). We note that the EU/FP5 FLOSS developer survey of over 2800 respondents showed that a majority of developers themselves identify with the term "free software", while Libre software (logiciel libre, software libre, software libero) is the favoured term in southern Europe and Latin America.

under other permitted terms[4]. FLOSS has leapt to prominence by taking significant share in some specific segments of the software infrastructure market.

2 Background, objectives, and approach of the study

The overall objectives of this study are to address the following points and questions:

- What would be the potential impact on the development of the Information Society if public organisations (administrations, research institutions, universities, agencies, public companies etc.) were to release software fully owned by them under a FLOSS licence? Is an amplification effect on the adoption and use of information society technologies to be expected? What are the possible characteristics of such effect and under what circumstances would it take place?
- What are the conditions under which software fully owned by public organisations can be made available using a FLOSS licence? What are the opportunities, barriers and limitations to this process?
- What options and recommendations can be provided for actions that could be taken, notably by the European Commission, EU Member States, and the private and public sector in general?

We understand that our task is three-fold:

- Considering the principle, legal aspects of the question cannot be avoided: could the public sector decide to publish all its software under a FLOSS licence? Would there be limitations with regard to the national and European Community rules regulating fair competition, public procurement rules, or government security rules?
- Can the public sector, by publishing its own software under FLOSS licences, have a significant impact on the shape and character of the information society?
- The last aspect is of course to explore all tangible and practical aspects of FLOSS publishing: size of the community, perennial character of the solution and of the processed documents, perspective for the solution to become financially independent from the public purse, support, improvements, quality, security and standards conformity.

In order to provide meaningful answers to these questions that help to advance the public sectors in Europe, the study carried out quantitative and qualitative analyses of the possible impact on the development of the Information Society and present ICT take-up scenarios, based on realistic data, using indicators for the level of ICT adoption, and explaining the likelihood of the scenarios.

Our approach takes into account that releasing publicly developed software as FLOSS has both static and dynamic effects. Statically, it may have effects on competition. Dynamically, it may affect the rate of innovation and take-up in the information society. The study investigated this through a combination of empirical, quantitative and qualitative approaches, supported by economic modelling. For the study, existing data on FLOSS in European public sector has been re-evaluated, and new data on software release activities of public bodies has been generated.

The cases that have been analysed for this study are:

[4] The four basic freedoms of FLOSS are described at http://www.gnu.org, together with more information on FLOSS licensing. A more detailed definition of FLOSS by the Open Source Initiative (OSI) is available at http://www.opensource.org/docs/definition.php.

- London Borough of Camden (local): this public body developed under contract successive versions of a content-management system targeted at public authorities' needs. The system, called APLAWS, is published under the GPL, and has been adopted by other organisations. A short case study published on the Open Source Observatory (Mangham and Ghosh 2005) examines one aspect of this experience, relating to how the public body was able to issue a call for tenders for open source software development.
- Extremadura (regional): very well publicised but not very well studied, this European Regional Innovation Award-winning instance of usage and development of open source deserves attention for its comprehensive socio-economic and organisational approach. The regional government of Extremadura distributed a localised version of GNU/Linux, promoting digital alphabetisation as well as the development of the regional IT sector.
- MMBase (multilevel – international/national/local): MMBase, a sophisticated content management system (MultiMedia database) was developed by VPRO, a national broadcaster in the Netherlands. Five years ago, it was released as open source. First other broadcasters, then the City of Amsterdam, followed by other public bodies in the Netherlands, adopted the software and started contributing to development. A number of businesses, start-ups in particular, support the development now, and together with individual developers and (often public sector) users formed the MMBase Foundation in 2002. MMBase is now the largest open source developer community in the Netherlands, and is used and developed on hundreds of large websites around the world, with deployment as far away as China.
- Beaumont Hospital is the largest teaching Hospital in Ireland. It experienced a severe cut in its IT budget in 2001 and reacted to this by migrating completely (including desktop applications) to FLOSS. After the migration, the IT department began to customise FLOSS to the specific needs of the hospital. The next step was development of new software.
- NFI, the Dutch Forensic Institute, has developed a number of FLOSS products that are specifically used for forensic purposes, such as TULP2G, a software that allows to read out data from mobile phones. The NFI has a long tradition of developing (proprietary) software and decided to release as much software as FLOSS as possible. This has resulted in faster release cycles for software developed by the NFI.
- CommunesPlone is intended to help municipalities to become independent from software vendors by contracting out the development of software that is needed for public administration, and to share it as FLOSS with other municipalities. The software is based on Zope and Plone. CommunesPlone encompasses a network of 14 Belgian and French municipalities and an international network of SMEs. It is intended to grow into an international network of public sector organisations. The project's intention to build an ever-growing network of public sector users of the shared software has been successful at a very early stage.

Based on the results of the quantitative and qualitative studies, combined with the economic modelling, the project consortium develops scenarios for the impact of public organisations' contributions to FLOSS on the development of the Information Society. Whether and to what degree FLOSS from public organisations contributes to standard general purpose application domains, embedded in the wider FLOSS community, or to what extent they contribute to niche application domains and rely on in-house capacities, is among the starting points for these scenarios.

The opinions expressed in this Study are those of the authors and do not necessarily reflect the

views of the European Commission.

3 Incidence and role of FLOSS in European local governments

As the two OSOSS surveys for the Dutch government (Ghosh & Glott 2003, Ghosh & Glott 2005a, Data Source S5) and the EU-wide FLOSSPOLS local governments survey (Ghosh & Glott 2005b, Data Source S4) showed, the use of FLOSS in public organisations concentrates on operating systems (GNU/Linux), Internet applications (Apache PHP, mail servers), and database programs (MySQL), with the desktop suite OpenOffice playing a smaller but still noticeable role (see Figure 1 below). There is only one group of programs that reach usage rates that are comparable to the rates of Linux and Apache, which is the residual category of "other programs". This category comprises niche applications that serves the purposes of specialised tasks for which public organisations are responsible, but which probably have no potential as packaged (or standard) software products on a wider market. We therefore expect that public organisations that already are or will in future become active FLOSS producers will either focus their activities around standard products that are already on the market and for which extensions, improvements, or new functionalities will be developed, or develop new software products for niche markets particular to the public sector.

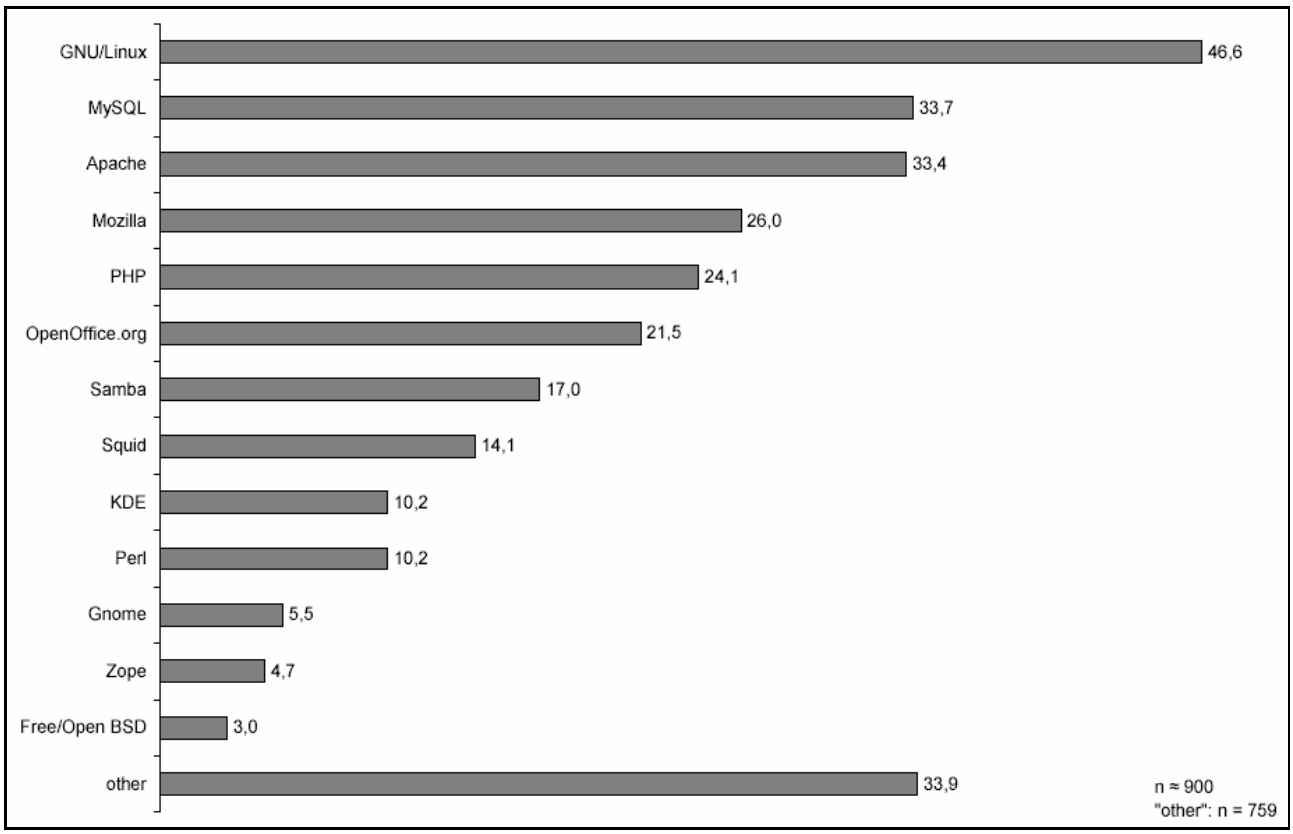

Figure 1: Incidence of specific FLOSS applications in public bodies, % of respondents

Copyright © 2005, MERIT. Source: Ghosh & Glott 2005b.

The FLOSSPOLS government survey revealed that awareness of FLOSS leads to increased use of FLOSS, regardless of whether or not an organisation actually uses FLOSS. Unaware users of FLOSS show stronger reluctance to FLOSS than those who know clearly what systems are used in their organisation and who have a clear attitude towards or against FLOSS.

As a main obstacle for the usage of FLOSS the FLOSSPOLS government survey identified fears concerning lack of technical support for FLOSS systems and high training costs in the course of migrating towards FLOSS systems in the organisation.

Budget cycles can become a hindrance towards FLOSS usage because a good part of migration costs (such as training costs) occur in the first year of the migration process, whereas savings tend to occur later in the course of the migration process. Since budgets are usually calculated for each year it is difficult for the IT department to convince other departments from the usefulness of a migration. A solution to this problem could be to schedule budgets for a period of three years, which would probably make migrations to FLOSS easier.

In our surveys, of the ten per cent of respondents who own software they have or plan to release under a FLOSS licence, there are two main motivating factors. One relates to the FLOSS community and philosophy, and can be considered an external motivating factor. The second relates to service quality and can thus be considered an internal motivator (see Figure 2).

Figure 2: Reasons for public bodies to release software under an OSS license – survey results

It should be noted that half of the software-releasing respondents checked both the first two reasons[5]. This might imply that the motivation of local governments to release their own software as FLOSS is to a high degree determined by this two-fold (internal-external) orientation. This finding, assumed it can be validated in further research, implies that the willingness of public bodies to publish their own software as FLOSS software might be as dependent on the situation of and services provided by their organisation as on their involvement with and understanding of the

[5] However, due to the sample size this correlation could not pass statistical tests.

FLOSS community.

This observation is fully in line with our findings from the case studies: case study subjects were all familiar with FLOSS, its ideas and principles. This familiarity was usually provided by individuals in the organisation who, in a certain context of decision-making on the implementation of services into software, were able to convince the organisation as a whole to decide for Open Source.

Regarding the non-releasers (i.e. local governments that did not already and do not plan for the future to release software as FLOSS, there is only one relevant reason for not distributing own software as Open Source, and that is that they do not have the necessary capacities / skills within their organisation to realise such a project

Apparently, many public sector organisations see themselves as s service providers, but do not relate this to the opportunities provided by software development and distribution. With regard to software, they define themselves as users and consumers rather than as suppliers, and they do not consider software as an integral component of their services and the way they provide their services. Policy initiatives aiming at increasing the role of the public sector in developing and distribute their own software should address this self-understanding and the perception of the role of software for the provision of public services in the information society.

4 Challenges and advantages for public bodies of releasing their own software as FLOSS

Challenges:

- *Programming capacities:* An important precondition for public organisations to contribute to the provision of FLOSS is to have or build up their own programming capacities, which can be achieved by employing skilled in-house staff as well as by contracting external firms or individual developers.
- *Perceived lack of support for FLOSS:* Though 39% of respondents in our surveys considered it problematic to find technical support for FLOSS systems, this may be perception rather than fact. Our data shows that this sentiment is strongest among FLOSS non-users and those who are unaware of FLOSS. FLOSS users, who must actually bear the costs of training and technical support, slightly tend to disagree with the statements that training and support are expensive or hard to find.
- *Limited regulatory area:* While FLOSS development often draws upon the collaboration of people around the world, specialised public sector software may be specific to a regulatory area, e.g. a country. This might limit the interest of potential contributors from elsewhere. On the other hand, if a community develops around the software, it is likely to be a local one, concentrating the benefits (such as skills improvement).
- *Copyright, patent and trademark issues:* Even if a public body holds (as it should) the copyright to all the software it distributes as FLOSS, publication of the source code may increase the releasing body's exposure to lawsuits concerning copyright, patents or trademarks in the source code. While there is no possibility to guarantee that such lawsuits never occur, prudent handling of legal issues related to the software will minimise this risk to the greatest extent.

Advantages:

- *Cost savings through software reuse:* Public organisations making their own software available under FLOSS licences will contribute significantly to the pool of cost-efficient software of potentially high quality (and likely exhibiting high attention to user requirements, having originated in customised software for specific needs). This contributes to a pooling of resources among public bodies, potentially allowing significant cost savings as they are able to reuse software applications made for specific public sector requirements, rather than having to duplicate them for each public body individually.
- *Participation and influence in FLOSS development:* The FLOSSPOLS survey of developers, among other sources, shows that growing interest in and experience with FLOSS increases the likelihood of contributing to FLOSS development; so we believe it is likely that there is a great potential for public organisations, as leading users of FLOSS, to play an emerging role as providers of FLOSS with a significant impact on the development of the information society.
- *Various advantages from the use of FLOSS:* Cost effectiveness, improved flexibility, increased long-term control over maintenance and support, more relevant technical features, increased interoperability, and independence from software vendors.
- *Skills improvement:* Usage and development of FLOSS will result in improved skills among

the IT staff, providing the organisation with better-trained employees. This is conducive to greater flexibility and increased IT performance.

Cornelia Kutterer of BEUC, the European Consumers' Organisation,[6] was asked to provide the consumers' perspective on PAs making their own software available as FLOSS. She strongly supported view that economy and society – and thus citizens and consumers - would benefit from an increase of the share of FLOSS in the market for software: “OSS production in public administration is not a core issue of BEUC, but BEUC has a clear opinion towards this subject and thinks that there would be indirect positive effects for consumers if the public sector would play an active role in the provision and usage of OSS: In general, FLOSS helps to improve competition because it provides better products than proprietary software and it avoids vendor lock-ins. As a consequence, consumers would be provided with better quality and liberated from the market power of large software vendors. Governments and public administration should support usage and development in the public sector in order to increase network effects and interoperability. This does not mean that proprietary software should be replaced by OSS, but that FLOSS should generally be offered to the consumer to the same degree as proprietary software in order to interact with public administration.”

[6] BEUC has long participated in the political discussion on FLOSS and repeatedly pointed out that FLOSS provides positive effects on competition and growth and that there are severe risks aligned with the dominant market position of proprietary software. This subject remains an important topic for consumers' organizations and they will go on playing a role in this field. Most recently, the Trans Atlantic Consumer Dialogue (TACD) has discussed user perspectives on Open Document Format (see http://www.tacd.org/docs/?id=304).

5 Conditions and effects of publishing software under an OS license – theoretical considerations

In this section we develop some theoretical concepts which can be used to analyse the conditions under which public release of FLOSS might have strongly positive impacts on the software sector as a whole. The main issues revolve around costs and benefits of software development and use. The interplay among different costs and benefits is highly complex. There are many different types of actors involved: commercial software houses; FLOSS developers; government agencies; passive users of different types; active user/developers and so on. Development costs vary with the version; with the type of development (open or proprietary); the formal structure of the FLOSS community; with the extent to which the product is standardized or custom built. Benefits from use are equally complex. To answer the main question involves understanding all of these factors, and getting a grip on how they interact with each other. Consequently, in this section we do not attempt to develop either a full-blown formal theoretical model, or a completely specifiable econometric model. Either of these would be extremely large and complex, and therefore very hard to understand. Further, in the case of an econometric model, the extreme data demands would make a complete specification virtually un-testable, due to the absence of necessary data. Instead, here we will develop two analytical lines which serve to highlight two of the major features involved in understanding costs and benefits. The point is not, as we said, to create a full theoretical model, but rather to develop concepts that are both necessary and useful to understand the forces at play. The structures we develop capture the main forces at work, and provide the tools for understanding when release of publicly developed FLOSS will provide social benefits, either through decreasing development costs, increasing quality, or increasing utility in use. Our goal here is not to make innovations in modelling of industrial structure and innovation. Rather it is to use existing, well-established models, and the results from them, to shed light on the question at issue.

The theoretical structures developed here are based on two standard models of industrial organization, namely Cournot (1838) and Hotelling (1929) models. These two structures are applied to the specific case of the private software sector and, in our opinion, they capture very well several of the key factors at stake. In particular both models describe a situation where the market is oligopolistic, which means that relatively few producing firms with high market power are present. Both also assume that marginal costs are positive and that they make up a non-trivial part of the total costs.

Before we go on, a bit of elaboration is in order, since this assumption seems to violate one of the received pieces of wisdom about software: namely that it has very high fixed costs and very low marginal costs. The former stems from the high development costs of any sophisticated piece of software, which is taken as the cost of the first unit. The cost of the second, and further units are merely the cost of electronic duplication (virtually zero) and then installation. Even being wildly pessimistic about the time involved in installation, the cost is dwarfed by development costs. This cost structure is due to the nature of the "technology of production" which allows high extensibility and perfect codification (Shy, 2001). Here, "production" indicates the duplication and installation of a fixed piece of software code, representing a single version of one product.

This traditional characterisation of the nature of software production does not always hold. In the first instance, firms producing packaged software have very different cost structures from firms providing custom software. The nature of the marginal costs of firms providing custom software or services depends highly on the rate of re-use. Firms may face high marginal costs depending on the difficulty of re-using users' specifications, designs, source codes and methodologies. This may be

particularly relevant if the firm is providing similar software (a database, say) to competing firms in one industry (insurance companies for example). Here a client of the software vendor may be very aggressive in preventing its competitors from "free-riding" on its purchase, due to code re-use. If code re-use is limited (either for legal or technical reasons) the second version of even a very similar product will not have zero marginal costs. Even in the case of packaged software, though, the zero-marginal-cost assumption may not be reasonable. Again it depends on a specific characterization of the product, namely that it represents a one-time purchase of a single version. Certainly, within a single version of a piece of software, marginal costs are very close to zero. This does mean that firms can capture and exploit a market for a single version. But if this is the only thing a firm does, it faces a problem for the future. Without further effort it will not attract the next generation of users. In addition, though, the firm will lose its original customer base, as hardware and particularly operating system software are upgraded, making the original version of the software obsolete or literally un-usable. Consequently, most successful packaged software goes through multiple versions, apparently endlessly in the strongest cases. The presence of network externalities, technological interdependence (between hardware and software), and consequent lock-in effects on the users' side is a strong argument in favour of an intense commitment from the producer side as well. A producer's goal is to capture generation after generation of users. But a user decides to adopt particular software if and only if he foresees a commitment from the producer to maintain and upgrade it up to some time in the future. In this sense, if producers want to exploit both network and lock-in effects they are likely to assure the upgrade and maintenance of the product, and from this perspective, the "second unit" can be seen as the second version, rather than the second copy of the first version.[7] This is not the typical way that goods are characterized, but by using this characterization we can deploy well-understood economic models to capture the effects that matter in this case.

There is empirical basis for the assumption that software for the public sector involves "small" but positive marginal costs. This means that firms present in the market should face a cost in providing additional units of a product. As we have argued in the preceding session, this is very likely to happen in the European software market for two orders of reasons.

First, the extent of custom software and services is almost the 50% of the market. This means that at least 50% of software is not similar to packaged software. It is not produce a standardized product with a one-time cost of production and near-zero marginal cost. Instead much of software results from firms trying to satisfy customers through ad hoc solutions. Hence, they are not selling exactly the same type of product to different customers, as producers of packaged software actually do. In this way, they adapt the product to specific users' needs and face, consequentially, small positive marginal costs.

Second, even if producers of packaged software are taken into account they are likely to present positive marginal costs as well. This is due to the strong commitment that producers must make to customers if they want to reap the profits of network externalities' advantages.

To prove these propositions we rely on the results from FLOSSPOLS and OSOSS surveys. The former was conducted on local governments in selected European countries in 2004 (FLOSSPOLS, 2004), while the latter was conducted in both 2003 and 2004 on Dutch PAs (OSOSS, 2003; OSOSS, 2004). The three questionnaires contain a set of similar questions to which respondents were asked

[7] In essence, what we are suggesting is that if a firm sells n copies of version 1, it is not the costs of units 2 to n which matter, but rather the cost of the n+1th unit, namely the first copy of version 2, which is crucial. From the point of view of social welfare, it is that cost which is vital, and which might be affected by the release of publicly-developed FLOSS. Treating a "version" (rather than a copy) as a unit of the good permits us to use standard assumptions about marginal cost, and thus use standard models from industrial organization theory.

to answer. In particular, we are interested on the questions regarding the extent of customization and the deployment of external maintenance. As Table 5 shows a percentage that ranges from 39% to 53% of PAs' interviewed have to customize their software after implementation on a regularly basis, while the same organizations rely intensively on external suppliers to do that (see Table 6).

Hence, under the reasonable hypothesis that PAs are important customers of private software firms, this points out the strong commitment software firms should make towards long term relationships with these customers. The fact that private firms are principal software vendors of PAs can be inferred by Table 7. Here the perceived dependence on private firms as software suppliers is presented. From 44% to 67% of respondents see their organization as highly dependent on suppliers. Furthermore, most of governments rely on less than 4 suppliers, meaning that their dependency is high, and that it reflects the oligopolistic structure of the market assumed in the theoretical model (see Table 8).

All these factors help explaining the nature of the marginal costs. In fact, they put pressure on costs, in particular marginal ones. So, not only the first unit of software is expensive to produce (as standard economic theory asserts) but also following units will have non-zero costs as it must be incorporated in the single unit the cost of maintenance to be provided – especially given our perspective in this model that treats the software version as the "unit".

Do you think your organization is too dependent from vendors/suppliers?			
	OSOSS2003	**OSOSS2004**	**FLOSSPOLS 2004**
Yes	67.3	67.7	44.0
No	32.7	32.3	48.6
I don't know			7.3
Total	100.0	100.0	100.0

Source: Our computations on FLOSSPOLS (2002), OSOSS(2003) and OSOSS(2004)

Table 1: Perception of dependence on suppliers

Table 6: Number of software vendors

Number of software vendors	OSOSS2003	OSOSS2004	FLOSSPOLS 2004
1 - 4 vendors*	52.3	26	52.5
more than 4 vendors*	47.7	74	47.5
	100.0	100	100.0
Total			100.0

Table 2: Number of software vendors

Note: 1-3 and more than 3 for OSOSS2003. Source: Our computations on FLOSSPOLS (2004), OSOSS(2003) and OSOSS(2004)

How often do you have to customize software after implementation?

	OSOSS 2003	OSOSS 2004	FLOSSPOLS 2004
never	8.4	11.2	10.2
sometimes	40.4	36.1	51.0
regularly	41.6	36.1	23.6
Often	9.6	12.2	8.2
very often	..	4.4	4.1
I don't know	..	..	2.9

Table 3: Frequency of customisation after implementation

Source: Our computations on FLOSSPOLS (2002), OSOSS(2003) and OSOSS(2004)

To what degree do you deploy external maintenance services?

	OSOSS 2003	OSOSS 2004	FLOSSPOLS 2004
Never	3.9	3.4	7.2
sometimes	41.3	34.5	44.1
Regularly	30.7	31.5	29.8
Often	12.8	18.2	8.0
very often	11.2	12.3	8.3

Table 4: Degree of external maintenance services deployment

Source: Our computations on FLOSSPOLS (2002), OSOSS(2003) and OSOSS(2004)

5.1 Producers

Thus, assuming small but positive marginal costs, the standard Cournot model, which has been usually used to study traditional sectors, can be extended to the analysis of software.

The main hypotheses we use are:

1. There are several (but relatively few) firms producing similar, but not identical, products. This implies that firms have a certain degree of market power. In addition, since products have similar characteristics users can easily substitute among them without experiencing a decrease of their level of utility. This is a rather strong assumption that will be relaxed once that Hotelling model is introduced.
2. Firms do not co-operate. This means that firms are not allowed either to co-operate, for example by setting a market standard, or to collude, for example to prevent entry into the market.
3. The number of firms is fixed. This assumption will be relaxed in the discussion where we consider entry.
4. Firms play a two-stage game where they first choose simultaneously the quantity to produce and then, knowing each other's capacity, they simultaneously choose price.

This describes a standard model, well-known in the industrial organization literature, namely the Cournot model of oligopolistic competition. Here we highlight several results that are useful for us:

- The prevailing market price is lower than the price obtained in monopoly and the aggregate profit is lower than monopoly profit;
- The prevailing market price is higher than the perfectly competitive price and the aggregate profit is higher;
- As the number of firms in the market increases, both market price and each firm's profit decrease. At the extreme, as the number of firms gets very large, the market approximates a competitive market;
- The market share of any firm is negatively related to its marginal cost of production.

In order to understand the dynamic followed by market shares of firms in the software sector and the rate of competition and innovation of the sector it is important to understand factors affecting marginal costs of production of these firms.

The costs or production for any firm can be seen as affected by three main factors:

1. Technical difficulty in producing the software. This refers to the number and types of features the software has. Obviously, software composed by complicated and numerous features will require a high and intensive effort for maintenance and upgrades by the producing firm. In such a case the marginal cost (for each additional version) will be higher than for simpler products.

2. Competence of the producing firm. Inputs to software production are largely human capital. If a firm has a highly skilled labour force, it will spend fewer person-hours to produce a new version than will a firm with a less-skilled (or less appropriately-skilled) labour force. A firms privately-held knowledge assets are an important determinant of its costs.

3. Public knowledge available to the firm. Public inputs play an important role. In this respect, publicly available standards or a common knowledge base are all important factors. Furthermore, the presence of an important base of FLOSS, which a firm can either use outright or can use as a base on which to build, is a factor that affects positively the stock of public knowledge available to the firm. We would expect that firms in a software sector with much available relevant public knowledge will have lower costs than firms in a sector having little available public knowledge.

It is important to point out the difference between R&D spending and innovation. A reduction in R&D spending may not necessarily reduce innovation if, for example the reduced spending is used more productively, or if there is a bigger platform of public knowledge which can be leveraged by private firms. (Indeed one could argue that the presence of a larger public knowledge base makes private R&D spending more productive.) Similarly, increases in R&D spending may not increase innovation, particularly if that innovation relies heavily on publicly available knowledge.

With this background, we can turn to the effects that the release of software by public bodies will have on the private software sector. In particular, our concern is on the effect that the release of FLOSS code held by public organizations will have on the profitability and the competitiveness of firms operating in the European private market. European public administrations have been quite interested in FLOSS in recent years. There many public administrations now develop their own code internally. This means that there is a large code base which could be released for free. If a decision is taken at this regard, then its impact on the private software sector should be understood. In particular, the effects that this decision is likely to have on firms' profit structures and on their rate of innovation are surely relevant.

To begin we ignore the possible effects on entry to the market, and look only at the effects on existing firms, assuming that no new firms enter as a result of FLOSS release. Under this condition, the release of FLOSS is likely to change the variance in the cost structure of firms. After release, all firms will have access to a larger stock of public knowledge. All else equal, this implies that they have a more similar mix of inputs than they had before the release. This implies a reduction in the variance of their costs. Since market shares are determined by costs, according to the Cournot model, and profits are determined by market shares and costs, we observe a follow-on effect of reducing variance in the profits of firms. Market share, market power, and profits of the large firms will fall, those of the small firms will rise. Where this has a dynamic effect is in R&D expenditure. In the software industry of established firms, R&D spending tends to be financed by retained profits, and indeed there is evidence that firms tend to spend a fixed proportion (around 15%) of their profits on R&D (Source: Abi-Saad P., David C., Gandon M., Weisenburger E. (2001), Recherche & Developpement en France: Resultats 1999, Estimations 2000, ObjectifsObjectifs Socio-Economiques du BCRD 2001, Paris, Ministere de l'Education Nationale.). Decreasing market shares will mean decreasing profits for larger firms, and hence decreasing resources to be devoted to R&D activity. The opposite happens for smaller firms that have more money for the R&D process.

This effect on the return on investment in R&D can be understood using Figure 3. A non-controversial assumption that R&D has a decreasing marginal product suggests that this re-distribution of profits is a good thing. Larger firms, having more profits, make larger investments in R&D. The marginal return will be small, so reducing their profits and so reducing their R&D expenditure (from *h* to *g*) will have little effect on industry innovation (segment *ab*). Smaller firms, with small profits and small R&D expenditures have a higher marginal return to R&D (segment *cd*). Increasing their expenditures (from *e* to *f*) will increase their innovation by segment *cd*. Since *cd* is larger than *ab*, total industry innovation increases.

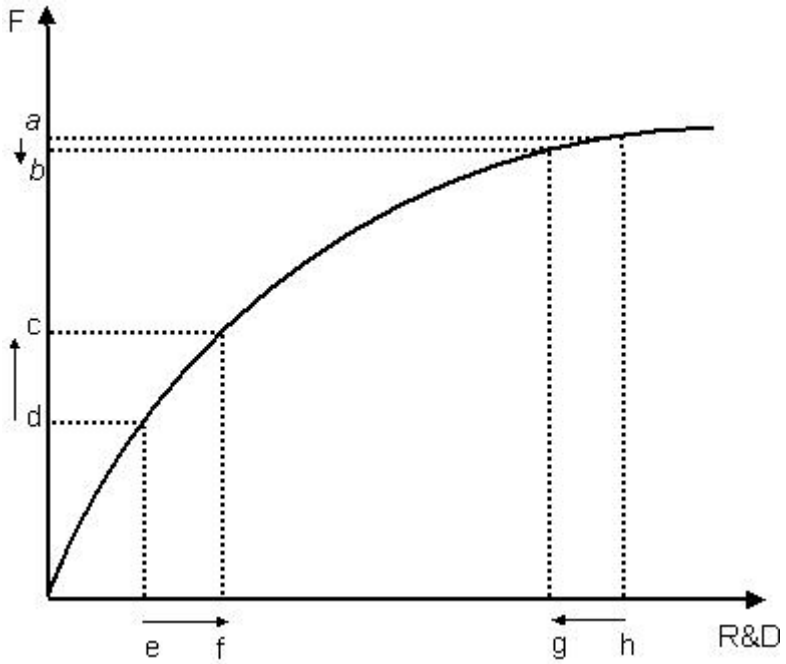

Figure 3: Marginal Return of R&D by firm's size.

A shift of profits from high-profit to low profit firms will, all else equal, improve the industry-wide

returns to R&D, and so increase industry level innovative performance. But so far we have ignored the effect on total industry profits. With a decrease in the variance of costs, total profits are likely to fall. Thus there are two counter-valuing forces: reduced variance in costs reduces the variance of profits, and thus R&D and so improves industry R&D performance; but reduced cost variance lowers industry profits, reducing industry R&D, and so reducing innovative output. Which effect dominates cannot be judged a priori. However, the effect is most likely to be positive in sectors in which the variance in firm size is large, particularly if the largest firms are very large and so well into severe decreasing returns to R&D spending.[8]

So far we have also assumed that no firms enter the market in response to an increase in public knowledge. If this is relaxed, and entry into the market is allowed, other effects come to play an important role in order to understand the possible scenarios following a release of FLOSS code by public bodies. In this regard, the nature of the firms entering is extremely important. If the majority of firms entering are large companies then no efficiency gain will probably take place, while with small firms entering the efficiency gain will be higher. At the same time, regardless of the size of the firm, the entrance by new firms into the market is likely to produce a "threat effect" on incumbents. This is due to the fact that they fear new entrants since more the presence of more firms tends to reduce the profits of existing firms. Thus, incumbents are likely to reduce dead-weight losses due to monopoly power by themselves in order to compete effectively with new entrants (Gilbert and Newbery, 1982). Thus the threat of entry, initiated by the release of FLOSS code, will reduce monopoly profits, and constitute an immediate gain for the consumer. However, the reduction of profits may imply a reduction of R&D, creating a longer term slowdown in the rate of innovation. Again, it is exceedingly difficult to weigh the balance of these effects a priori.

Finally, we must address one dynamic aspect of R&D. R&D has two functions: it provides immediate innovation; and it provides learning on which to base future innovation. A reduction in R&D by a firm today may reduce its ability to undertake productive R&D tomorrow. Again, this effect is likely to be governed by decreasing marginal product (it becomes harder and harder to learn, the more learning one does) but this may operate differently than the decreasing product in the innovation aspect of R&D, and so there may be a tension, again, between the short run and the long run However, while this "learning-to-do R&D" effect no doubt exists, it is likely to be significantly smaller in magnitude than the previous effect of decreasing marginal product in innovation, particularly in a field such as software, where all firms that survive must have high abilities at R&D to begin with.

According to this dynamic interpretation, the increase in the public knowledge input consequent to the release of OS code by public bodies presents a trade-off. On one side, the availability of this knowledge stock base for free fosters innovation. On the other side, the effect on profit structure is likely to diminish the R&D spending both at the firm and at the industry level reducing the rate of innovation.[9] A graphical description of this argument is shown in Figure 4.

[8] Anecdotes from the software industry in the 1980s and 1990s describe the R&D spending by Apple and Microsoft as "burning money" it was so large. If true, this was a case in which a reduction in variance of profits among firms would certainly have improved industry innovation performance.

[9] For an analysis on the relationship between market structure and R&D spending in a dynamic setting see Cowan (2002). There, today's R&D productivity is formally related to its future's productivity. The final result underlines how, at the firm level, the number of firms in the industry are negatively related to R&D. On the contrary, the relationship is not so clear at the industry level where it can be both positive and negative depending on specific parameters (rate of knowledge accumulation, success probability of innovation processes, etc).

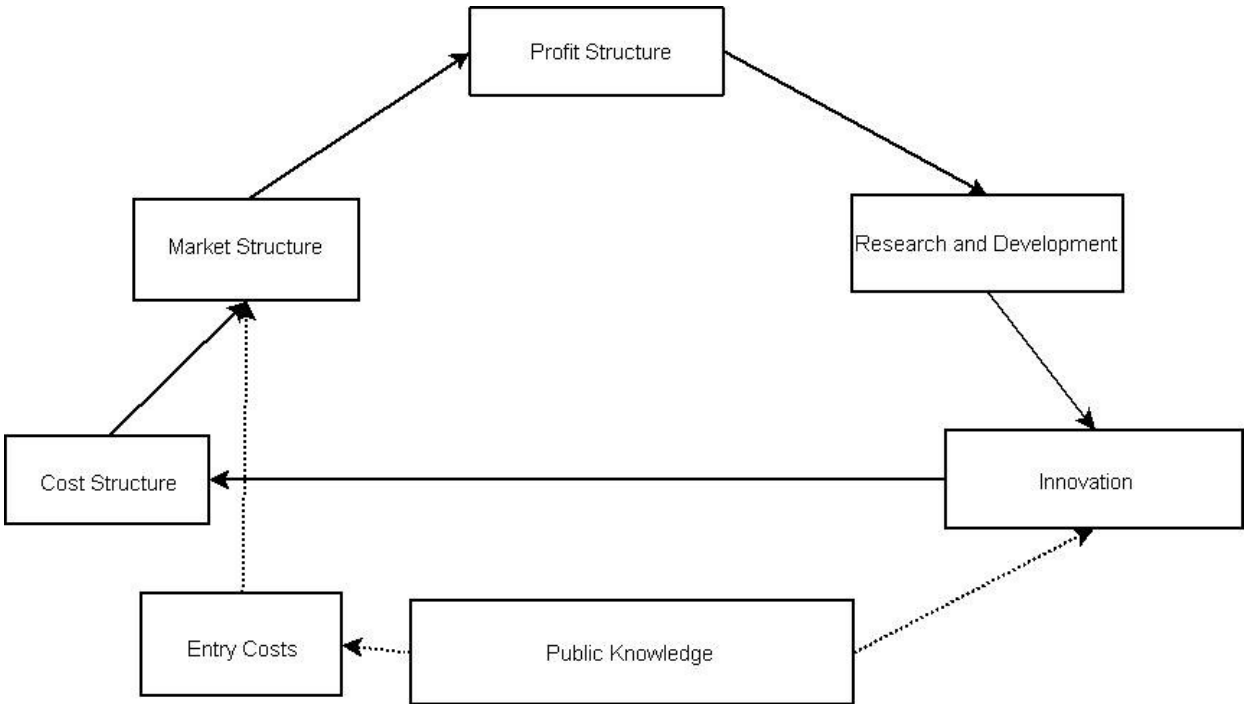

Figure 4: Effect of the release of FLOSS code by Public Administrations.

5.2 Users

To this point we have focussed on the supply side of the market. A release of FLOSS has effects on the cost structures of producing firms. This in turn has an effect on the competition in the market, changing market shares and profits. The link between profits and R&D drives these effects through to have an impact on the level of innovation in the sector. But this says little about the effect on consumers.[10] We turn to this issue now, drawing on a different type of model. Here we acknowledge that not all software users have the same tastes, or want or need to do the same things with their software. Indeed users can be very heterogeneous, and one of the arguments against standardization is that any dominant standard will fail to serve some consumers well. How well users are served by the software products available is the focus of the next section.

Compatible with the model presented so far but introducing user heterogeneity, is the horizontal differentiation model. Originally, the first model of this type was developed by Hotelling (1929). This model, in its original form, takes into consideration the possibility for firms to locate in different places in order to profit from the different position of users. A more recent interpretation of this model interprets the location of users as relating to the tastes that they want to satisfy. The main result derived from the model is that each firm differentiates from its rival in order to soften price competition and obtain some sort of monopoly power.[11] If we imagine that consumers are located along a line representing their tastes or requirements (the extent to which they use software to draw pictures, for example) firms will spread themselves equally over this line.

[10] A reduction in profits generally implies a transfer of surplus from producers to consumers as the same product mix is available at lower prices. As stressed above, this is a short run phenomenon.

[11] In the localization interpretation, each firm decides to locate enough far from its rival in order to avoid price competition. Instead, in the product differentiation case, firms produce goods as differentiated as possible.

Each software product comprises a unique set of features. For the same of simplicity, suppose we can represent these features as being at a point on a line.[12] For example, in Figure 5 the *x* axis represents product space, i.e. the different types of software products available in the market. Different products satisfy different consumers' tastes.[13] Obviously, points close to each other represent products that would be tailored to similar preferences.

On the *y* axis the utility for the consumer is shown. Assume that only a few firms are producing software products. If a consumer is located at the same point is space as a firm, the consumer receives high utility from the product. On the other hand if the users are far from any firm, he must select one product, then either expend resources customizing the software, or simply be dissatisfied with the performance of the product, but have to live with it. Suppose there are three firms, located at A, B, and C. A user located at a will choose the product of firm A, and receive utility *u1*. The tent-shaped straight lines represent the utility from a particular product received by a user at any point on the line. Obviously, firms will produce products whose characteristics overlap which means that they will compete for consumers that are interested in them. This case is graphically displayed by the portion of x axis under crossing lines, e.g. segment bc.

If we assume that a software sector has few producing firms in it (A, B, and C) as the one presented in the figure, then the variety of products present in the market and, as a consequence, the level of utility for consumers can surely be increased by the entrance of new players. As we argued above, publicly available knowledge or inputs reduces costs. It equally reduces barriers to entry. Thus release of code would encourage firms to enter this market. They would enter at D and E.[14] By entering they increase the level of competition, since there are more segments on the product space that are located under intersecting lines. Before the release of the code, a consumer located at a purchased good A, and realized a level of utility equal to *u1*. After firms D and E enter the market, that consumer switches to good D, and his or her utility rises to *u2*.

[12] If the software had one feature: calculating square roots, for example, a point on the line might represent the ratio of accuracy to speed of calculation. Different users have different preferences for accuracy over speed, so they too would be located at a point on this line.

[13] Many different types of products are present in the software market. A main distinction in the classification is between software packages and customized software and services. Of course, the variety in users' tastes is easy to understand in the practice: some users prefer quite standardized products for normal operations (operating systems, enterprise resource planning, payroll/personal management, etc). On the other side, some other users have particular needs that are satisfied through custom software (custom software development services, systems implementation, etc).

[14] New firms position exactly in between other firms. In fact, this is the only way to maximize the monopoly power they have on a part of the product space. Indeed, this is the main result of the Hotelling model.

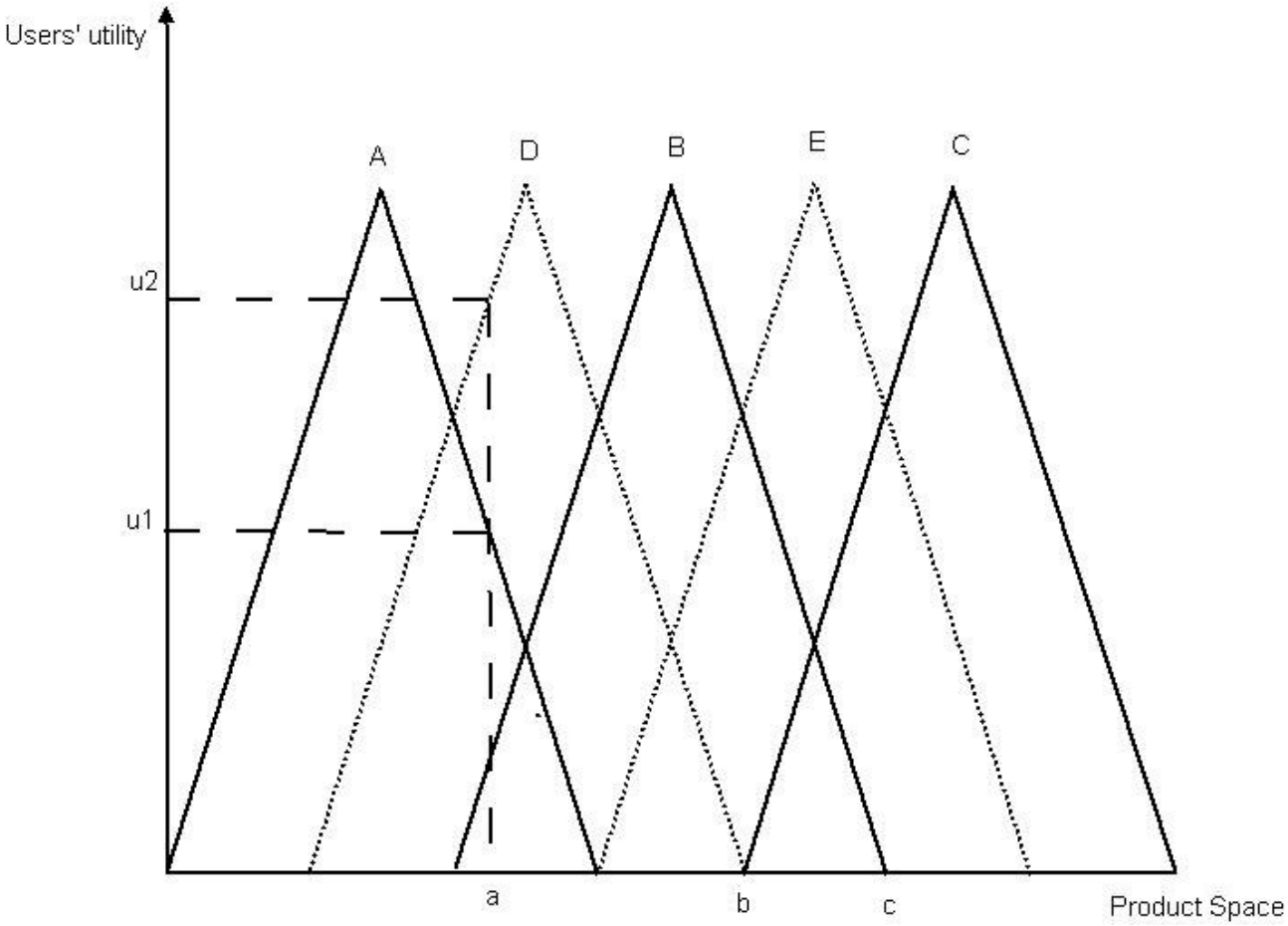

Figure 5: Effect of code release by Public Administrations in a horizontal differentiation setting.

A more careful interpretation of Figure 5 leads us to see when this effect is likely to be strong. The entry of new firms (D and E) is valuable because some users (such as a) are better able to have their needs fulfilled by one of the new products. Thus what is crucial in determining the size of this effect is not, in fact the number of incumbent firms, but rather how the location of the incumbent firms, in technology space, relates to the location of users in that space. The Figure implicitly assumes that users are evenly spread over the line, so an entrant at D will serve many users well. But if users are in fact bunched in their needs, and are mostly located at directly under A, B, and C, an entrant at D will have only a small impact. This suggests that lowering entry costs by releasing software as FLOSS is most likely to have a big effect on a market segment wherein there are few firms, but many users, and more importantly, many different types of users. Niche market software is not very likely to satisfy this criterion; the more general purpose the software, the greater the potential impact.

Network externalities

The analysis so far has ignored effects like network externalities on both supply and demand sides. Software is subject to strong lock-in effects, often driven by user learning: a user develops large amounts of knowledge regarding how to make the software perform the tasks he needs; and through

legacy: switching software applications typically demands translation of data files and often communications protocols and settings. This and other characteristics of software use present a large potential for lock-in, which will favour the technology that has first reached a defined installed base.[15] This is why, on the supply side, firms are competing strenuously both to have their products become the standard and to obtain a large installed base of users. This is usually done using low penetration prices and announcement of products before they are commercialized. If a user is locked in, the appearance of an entrant (D in Figure 5) does not help—even though the new product fits better the user's needs, it is too costly to switch.

The release of FLOSS code can play into this issue as well. The difficulty of switching can be characterised as an issue of standards. If file formats, communication protocols, user interfaces are standardized, switching costs fall dramatically. If switching costs are low, barriers to entry are low, since it is easier for a new entrant to attract users, and thereby build an installed base (and thus sell more copies of each version of his product). To the extent that released FLOSS is taken up by different developer firms, this represents at least some degree of standardization, and thereby lowers switching costs, barriers to entry, and can lead to an increase in user utility. Again, it is relatively easy to see where this effect will be strongest. If the released code is particularly good in terms of user interface; file formats per se; file format translation; or any other feature that contributes to user lock-in, the released FLOSS can contribute to the development of open standards, which are well-known to mitigate lock-in effects.

As a sort of corollary to the above line of argument, we mention effects of standards more generally. At the heart of network externalities lies the importance of standardization in ICT markets. The choice of a particular technology to be adopted by everyone has both positive and negative aspects. In particular, standardization helps firms avoid excess inertia (the prolonged reluctance to adopt one technology due to the high losses the firm can incur if another technology results as the winner) and it also reduces users' search and co-ordination costs. On the other side, lock-in to an inferior technology and variety reduction are negative aspects of the phenomenon.

In this regard, introduction of new public knowledge through FLOSS code early in the development life cycle, while a particular type of application has not yet matured, is likely to have different effects. A positive effect can be the higher degree of interoperability that can be obtained thereby. Indeed, the possibility for private firms to have open code lines for free would mean an increase in the possibility to interconnect their products. A higher degree of interoperability between systems and applications and between applications themselves is likely to take place, and would reduce the effects of lock-in generally. Furthermore, an increase in the probability that some inferior technologies would be unlocked and an increase in product variety are all effects that are likely to occur.

On the other side, an increase in search and co-ordination costs is a possible effect on the demand side, while on the production side excess inertia by firms is likely to happen. Users and firms could face less certainty about which product to adopt, resulting in higher opportunity and transaction costs. This could be ameliorated through the wider spread of diverse skills, and is likely to be more than compensated by the lower costs resulting through competition.

[15] For a formal presentation of network externalities see Farrell and Saloner (1985), Katz and Shapiro (1985). Arthur (1989) for a treatment of lock-in under increasing returns. For an historical account see David (1985).

Framework type	New firms' entry	Firm type	Positive effects	Negative effects
Producers (Cournot model)	No entry		Efficiency gain Profit shift from large to small firms Increase in total innovation due to re-distribution of profits	Decreased total profits and decreased R&D expenditure may lead to decreased total innovation
	Entry	Small	Efficiency gain Threat effect	Smaller monopoly profits implies less R&D expenditure; possibly less innovation
		Large	Threat effect lowers consumer prices	Smaller monopoly profits implies less R&D expenditure
Users (Hotelling model)	Entry		Lower prices Better match of needs and software features	
Network externalities	Entry	Small Large	Higher degree of interoperability Unlocking inferior technologies Increase in product variety	Increase in users' search costs Increase in users' co-ordination costs Excess inertia

Table 5: Effects of the release of FLOSS code by public Administrations under different conditions

Note: The table lists possible or probable effects. The relative sizes of these effects (and thus the net effects) will vary depending on the conditions under which software is released in each instance.

Arguments against the release of FLOSS by public bodies typically take the approach that public bodies releasing software are somehow distorting the market, e.g. by providing a subsidised (and unfairly competitive) product, thereby reducing profits, R&D spending and innovation (e.g. Evans and Reddy 2003). This simplistic view of the software market ignores the fact that even packaged software is usually not sold as a one-off product but as a bundle of many conditions and expectations, as described above, and it is the profits and R&D spending in the overall market that drive innovation.

Moreover, the majority of software (and thus most likely innovation) is not produced in software companies. In fact, various statistics show that among the biggest software producers are in fact software users creating custom software for their own needs: in-house or "own account" software estimates vary between 20-40% of the whole software production market in terms of value and employment, and a further 40-50% of the market is custom software (tailored to user needs by external businesses).[16] This is essentially a services business, as it involves customisation and, usually, on-going maintenance. Any market distortion provided by public bodies releasing FLOSS

[16] see, e.g. US Department of Commerce 2000, which lists figures based on labour statistics and census data, showing that packaged software has a 24.7% share of the total US software market excluding in-house software; Parker, R. & Grimm, B. 2000; Parker, R., Grimm, B. & Wasshausen, D. 2002; and Hermans, T., 2002 for a European perspective.

would be limited, however, to the 10-30% of the market represented by packaged software products.

In considering the impact on growth and innovation, we must note that “free resource” of software released by public bodies is available to the entire software market (including services, “own-account” in-house production, custom software, and to a limited extent even packaged software, based on the licence chosen for release).[17]

[17] Release under a reciprocal licence such as the GPL makes it somewhat difficult to incorporate the software into another packaged product, but doesn’t limit the potential for building packaged software products over the released GPL software.

6 Opportunities and barriers of publishing software under a FLOSS licence

As outlined in chapters 3 and 4, a number of issues come into play when PAs have to decide on software procurement and to involve or not to become involved with FLOSS. These issues bear positive aspects as well as deterring factors. For instance, advances in process organisation might be obtained through a loss of jobs. In the following sections we discuss the relative importance of each of these issues and analyse them with regard to their motivating and hindering factors.

6.1 Economic factors (cost-effectiveness, spill-over effects, employment effects)

6.1.1 Opportunities

Cost savings are usually the first item that comes to mind when reasons for a PA to use or develop FLOSS are considered. The importance of cost issues for the software usage in the public sector has been shown by empirical studies (e.g. Wichmann 2003; Ghosh & Glott 2005a,b; Ghosh et al. 2006), and cost issues play also a key role for software production as well as for software usage in our theoretical model (chapter 5).

However, it turned out that cost issues played a role only for three out of the six PAs that were analysed in our case studies,[18] with varying degrees of importance and different meanings of cost: The provincial government of Extremadura initiated GNU/LinEx because the government didn't have the budget to equip schools with the necessary number of PCs and software to achieve its infrastructural and societal goals using proprietary software. Thus, for GNU/LinEX cost were a very important motivating factor. The same applies, probably to a much stronger degree, to the case of the Beaumont Hospital in Dublin, where the decision to use and develop FLOSS was exclusively made because of IT budget restrictions. However, cost didn't play a comparable role for the Council of the London Borough of Camden in its decision to develop and distribute APLAWS as OSS. They were a motivating factor in the sense that the Office of the Deputy Prime Minister (ODPM) required to develop software solutions that can be distributed at low cost. A representative of the APLAWS project pointed out very clearly, apart from this requirement cost didn't play a role because the ODPM provided a budget for the project. Nevertheless it must be noted that without this additional budget the council would probably not have engaged in OSS.

Two lesson can be learned from these observations: One is that cost issues do not play a motivating role for all PAs when a decision on FLOSS development and distribution is made. The other one is that if they work as a motivating factor this is either due to great economic need (because of tight budgets) or because of inertia (with regard to changes in the software policy in a PA) because of a lack of additional financial means to finance such a change.

In our theoretical model, network effects play an important role with regard to advantages of FLOSS development and distribution by PAs as compared to purchasing proprietary software. One of the advantages of collaboration in networks is the opportunity to externalise cost that had to be borne by the PA's own budget if all development, maintenance, training, etc. would be performed in-house. Network effects and opportunities to externalise cost also exist in the private sector when proprietary software is developed and maintained. Yet FLOSS provides the opportunity to build up

[18] A full description of the case studies is provided in D3.

user/developer-communities, as we found in all our case studies. The advantage of user/developer communities is that many tasks like updating, adaptation, maintenance, etc. do not have to be contracted out to a company, but is done at comparably low cost by those who also use the systems.

Indeed, two PAs in our case studies (Council of the London Borough of Camden and CommunesPlone) reported that some of the costs associated with developing and distributing FLOSS could be externalised. Though we observed development activities outside the PAs in other projects, too – for instance MMBASE was not only developed by VPRO software developers but also by developers of the City of Amsterdam – it must be noted that *in the perception* of these PAs they have to pay for all development activities and services they need for the software system to meet their demands. Therefore we like to conclude that apparently many PAs have not yet discovered how to take full (or at least better) advantage of the network opportunities that FLOSS provides.

6.1.2 Barriers

Though our theoretical model argues that FLOSS is generally cost-efficient, cost can also work as as a barrier towards the development and distribution of FLOSS by public administrations. This happens, for instance, when the persons that decide on software purchases and the general IT policy of an organisation have a negative perception of FLOSS and fear that migrating towards FLOSS or engaging in FLOSS development projects would consume too much of their resources. Research has shown that these individual perceptions, especially of the share of license fees in the overall IT budget, have a significant impact on software purchasing decisions, whereas the actual size of the IT budget and share of license fees has no such impact. What is decisive is not that the share of license fees is 10% or 60% of the IT budget, but that the person responsible for software purchasing decisions considers this share as “too high”. Thus, in one case a 60% share of license fees in the overall IT budget might be perceived as reasonable and therefore has no consequences for software purchases, in another case a 10% share of software license fees in the overall IT budget might be considered as too high and thus initiate a migration towards FLOSS. (see Ghosh & Glott 2005b)

Our survey was designed in order to reveal insights in the role of such fears. It turned out that neither those PAs that have already distributed software under a FLOSS license nor those who did not are affected by fears of cost (only 14% of non-releasers and 10% of releasers expressed such fears). Thus, fear of cost due to a lack of knowledge of FLOSS do not deter PAs from developing and distributing OSS.

The case studies were examined with regard to the actual cost that occurred in the course of their FLOSS projects. As described in the case study analysis (see D3), almost all PAs that were scrutinised had difficulties to quantify the exact cost of their relative FLOSS project. All case studies reported though that if they estimate a rough balance they feel that they benefited from their FLOSS projects. These benefits were either obtained by savings on software purchases, optimised processing of services, by monetary benefits in terms of low cost for software licenses and training (as training costs could be reduced because Intranet and Internet use the same system), or by reusing software.

However, two cases reported cost-related features that seems to violate our model assumption. The Beaumont Hospital considered the benefits of re-using software (for budgets etc.) low, whereas our model suggests that re-usage of software provides competitive advantages. The crucial point regarding the software developed at Beaumont Hospital is that, according to the head of the IT department, the systems they develop are useful only in a hospital environment. Though in fact Beaumont also developed software that could be used by health organisations, this aspect deserves

some attention and will be discussed further in the following chapter on the impact that publishing FLOSS by PAs might have on the economy. As we will see, the degree to which such an impact can be achieved depends highly on the scope on which a software system can be used and re-used. It should be noted that our model includes this reservation, as it states that niche products might not result in comparable impact on market and benefits as general purpose software.

The other case that reported aspects that appear to violate our model assumptions is the Dutch Forensic Institute. The head of the software engineering group pointed out that there are no advantages of FLOSS over proprietary software in terms of R&D investments and returns of these investments. However, this statement referred to the efficiency of software development tools (which might not differ between FLOSS and proprietary software) and did not account for the cost that are aligned with purchasing these tools (which might differ a lot). Therefore we do not see a need to revise or specify our model assumptions in this respect.

6.2 Flexibility (vendor independence, control)

6.2.1 Opportunities

Empirical data (see chapter 2) as well as the theoretical model (here: from the perspective of the user) emphasise the importance of vendor lock-in for attitudes towards FLOSS and decisions on software purchases. But flexibility gains through FLOSS usage and development do not only derive from vendor independence. They may also stem from the opportunities to see how the program actually works (because the source code is visible) and to customise the software (because the source code can be changed). These different aspects of flexibility gains were strongly pronounced by four of our six case studies (London Borough of Camden, MMBASE, NFI, and CommunesPlone). The desire for flexibility gains is thereby not unique for the PAs that have been selected for the case studies, as indicated by our survey. It revealed that vendor independence and better customised software are needs that generally play a vital role for public bodies to publish their own software as OSS.

6.2.2 Barriers

In contrast to many other factors that are scrutinised in this report, flexibility does not feature an ambiguity of providing both opportunities and barriers at the same time. Therefore there are no barriers derived from flexibility to be discussed here. However, to take full advantage of the opportunities to gain flexibility might require a number of fundamental organisational and cultural changes within PAs, which are discussed under the respective sub-chapters of this section.

6.3 Legal issues

6.3.1 Opportunities

FLOSS provides such a broad variety of licenses, with more than 100 different licences available, that any PA that considers to publish its own software as FLOSS should be able to find a license model that meets its specific needs. Our interviews demonstrated however that the GPL licence is dominant, though other FLOSS licences are used, too.[19]

[19] In fact, GPL is so well known that everybody thinks that “if it is open source” or “if it is free software” a component or tool is certainly GPLed (we encountered competent lawyers thinking they would have to distribute according to the GPL because their solution was based on GPLed components – after investigating these components, we found

Since FLOSS requires the PAs to decide on licences and copyright issues, it provides the opportunity for PAs to generate new (legal) knowledge that was not available in the organisation before and that can be shared with other PAs. Indeed most of our case studies reported that their knowledge of intellectual property right issues increased in the course of their engagement in FLOSS development and dissemination projects. However, we found only one case (the Council of the London Borough of Camden) that considered itself to be able to give legal advice to other PAs, though with the important reservation that the quality of this advice might be questionable, due to the lack of experienced lawyers.

6.3.2 Barriers

On the other hand, the variety of licences makes legal issues associated with FLOSS very complex and difficult to oversee for PAs. The problem becomes even more complicated through the fact that some member state organisations are establishing national licences, seen as more suited to their legal framework (especially Latin law countries such as France, Germany, Spain etc.).

Given the fact that PAs often do not have disposal of legal staff that is experienced or even specialised in license issues, it should be a key task for PAs that engage in FLOSS development and publishing to find legal advice. Our interviews showed however that licence choice is generally done after consultations with internal staff. We found no case where external specialised lawyers were involved at the beginning of a project. Moreover, the choice is often made by developers and not by lawyers.

The involvement of lawyers only is however not always the best solution, because they will debate on the content of the licence (e.g. the re-distribution, warranty and liability exclusion conditions, applicable law, competent jurisdiction and the feedback to obtain in case of modification) and this could lead to specific choices (e.g. the MPL) or – worse – to the elaboration of a specific licence text, which would only reinforce the opaqueness of licensing issues.

In the current situation, the choice of the licence is often a historical choice that ignores the global impact on project community growth. This has sometimes considerable consequences, as in the case of the MMBase distribution, where the choice of an MPL licence still generates discussions on the legal frameworks that would ended three years before if a GPL compatible licence had been chosen.[20]

Some foundations have adopted the strategy to leave their copyright to individual authors from the supporting community, thus refusing to appear as “sole owner”: This applies to the case of MMBase, but it is hard to apply to other PAs where software has been made by employees inside their normal working mission under *responsibility* of the PA or was contracted out. In other cases, - and in particular when personal data are processed like in health systems – the law may require a responsible entity, and therefore the PA could not try to “hide itself” behind a FLOSS community.

Finally, a problem that was emphasised by two PAs (MMBASE and London Borough of Camden) is that it is sometimes difficult for them to keep control over how the program develops through development activities of actors that download the program, but are not closely (or not at all) related to the original user/developer network. The risk of a project to fork thus also exists for PAs, which might have a negative impact on standards and routines in public services (because they begin to

they were not distributed under GPL). The reality is different: while the GPL is indeed used by a wide majority of projects in number, a significant part of the most commonly used components are NOT distributed under GPL.

[20] Quotation from the MMBase CEO: “I'm pretty sure that if MMBase was brought under the GPL license, take up would have been much, much, much faster. Also because when you talk to governments, they know GPL. It's the only thing they know”

differ, depending on which version of the program is used).

6.4 Technical issues

6.4.1 Opportunities

Proprietary software is to a considerable degree a “black box” to those who use it. It is impossible for the users to see how the program code is written and how the program actually works. Neither is it possible to change the programs, e.g. in order to customise them to specific needs. FLOSS, in contrast, provides access to the program's source code, understanding of its functioning, and the opportunity to customise it.

Another advantage of FLOSS is that, due to the openness of the source code, the widespread usage of open standards, and network effects (as described in the theoretical model), FLOSS enhances interoperability.

6.4.2 Barriers

When PAs engage in FLOSS development and distribution, they are confronted with a number of technical tasks that are not relevant for PAs which purchase and administer software, and for which no resources exist within the PA. FLOSS development is characterised by network-based source code revision, code testing, bug tracking, communication with internal and external developers, communication with internal and external users, software build management, and performance profiling. It is therefore crucial for PAs to master these technological challenges.

There is a need to use and follow the coding conventions (i.e. GNU coding standards[21], Java code conventions[22], etc.). To ensure that the code can be distributed, extensively used and maintained by external organisations, APLWAS and MMBase put such code conventions in place. They disseminate the best practices inside their organisation to decrease the workload when someone other than the original author would like to update the code of the product.

Though code can be syntactically correct, it may not produce the expected result. Writing and executing test cases helps to discover such defective parts of code. All public bodies we examined make use of forums at SourceForge or similar institutions for this purpose. This aspect is further discussed under the section on organisational issues in this chapter.

A key measure of performance is user satisfaction, i.e. user needs impact the performance of the system. All case studies identified user groups or stakeholders which they considered to be relevant with regard to assessing their products' performance. Especially MMBase and the London Borough of Camden emphasised the role of their user groups in this respect. However, MMBase discovered the importance of integrating user needs in the development process only quite a while after the project had started, and the attempt of the management to take care of user needs was originally not supported by all developers. Some of the developers even opposed this attempt. Another example of the problems that may arise when user needs are ignored or incorrectly perceived is provided by the Beaumont Hospital, where severe cuts in the IT budgets led the IT department to realising a migration concept towards FLOSS that overstrained the capacities of the users. As a result, the migration of the desktop systems was stopped, and proprietary software systems had to be re-installed.

[21] GNU coding standards: http://www.gnu.org/prep/standards/

[22] Code Conventions for the Java™ Programming Language: http://java.sun.com/docs/codeconv/

Developers read the code and try to discover code parts that do not follow the code conventions or the design patterns. Code revision also implies the discovery of memory leaks and security breaches. In the case studies we found that this task was performed in-house (especially NFI) as well as contracted out to a company (e.g. London Borough of Camden). Finally, it is helpful to develop the application in a modular form to facilitate its maintenance, the so-called Model-View-Controller (MVC) design pattern. This point is related to that of the code conventions. Developers can take into account design patterns when developing a product. A design pattern is “a general repeatable solution to a commonly occurring problem”; this is slightly different than using code standard. Using such design patterns will help external developers to understand the architecture of the product and more easily find his way inside the code, whereas the code standard helps a developer to understand the code more easily. While GNU/LinEx, APLAWS, the software produced by the NFI and the software produced for the CommunesPlone project followed these coding conventions, MMBase had to overcome a number of problems that derived from the fact that the development process at the beginning was not well documented and did not follow coding conventions and modular design patterns.

In order to distribute a FLOSS product a lot of technical procedures have been put in place by our case study subjects:

- Writing and executing test cases to ensure that any upgrade, proposed by an internal or an external contributor, has no impact on the product behaviour. This is part of the reviewing process.
- Conducting performance testing to ensure that any upgrade, proposed by an internal or an external contributor, has no impact on product performance
- Usability testing of the products to ensure that any upgrade keeps the user interface of the product easily usable by a standard user
- Writing and correcting the documentation.

In general, the projects we examined have put in place the tools that allow them to produce a state-of-the-art FLOSS product. By default they had to set up a technical committee (see above) but also to set up all the tools to support the development like version control systems, automated software build, newsletters, a support website, etc. All the case studies have set up a project structure that aims to emulate basic structures of the FLOSS community in order to generate discussion about the project and provide a means for coding and debugging. SourceForge or similar forums play an important role in this respect.

APLAWS developers use “Instant Messenger tools to communicate and give some support to the geographically dispersed developers/users of their product”. Anyone who is interested in downloading APLAWS must register, which allows the project coordinators to keep track of the distribution and users of APLAWS. The same or similar tools are used by all public bodies. Direct (face-to-face) interaction plays also an important role, especially with regard to the integration of user feedback. This integration of users' feedback is organised in different ways. The London Borough of Camden organises APLAWS user meetings, Extremadura and MMBASE make ample use of their foundation (MMBASE) and website (Extremadura), whereas for NFI and Beaumont a very close direct interaction with the users is the primary way of feedback integration.

Configuration management is certainly a key concept when developing an OSS. It helps to exactly identify the code that is part of a build, in order to produce a certain version of a product. “It's a difficult task, certainly if your application can be updated by anyone in the world” says APLAWS and MMBase.

6.5 Knowledge creation and sharing

6.5.1 Opportunities

As pointed out in our theoretical model, releasing FLOSS through public bodies is aligned with a trade-off between R&D spending and innovation, whereby it is difficult to say a priori which effect carries greater weight: the positive effects of a greater knowledge base through publishing FLOSS, or the negative effects of decreased R&D spending. However, from the point of view of our case studies it is obvious that they benefited significantly from the positive effects of increasing and sharing knowledge. For instance, the representative of the NFI pointed out that developing and using FLOSS allows the release of results of R&D at NFI much faster than with proprietary systems, as the user and developer community can immediately have a look at the code or test the system and comment back to the NFI. The interviewee emphasised the willingness to share knowledge and the positive effects this would have on the product again: “The wider commercial and non-commercial community can take them (NFI's software systems) up and advance them.”

Due to the advantages FLOSS has in terms of interoperability, FLOSS allows easier exchange of data with other institutions. This point has been stressed especially by the NFI as well as by the London Borough of Camden. The latter provides also an example of how the scope of knowledge sharing and interaction between different PAs increases when they start to develop their own software solutions for the services they perform. The partners that collaborated in the project were not only able to agree on standards and design patterns regarding the technical issues (see the respective section in this chapter) of developing APLAWS but also on classification of services and processes in which services are performed (The Local Governments Category List - LGCL).

6.5.2 Barriers

Like flexibility, knowledge creation and sharing is not aligned with negative aspects that would hinder PAs to share knowledge, but it requires some cultural and organisational prerequisites that are discussed below.

6.6 Organisational and cultural issues

6.6.1 Opportunities

Developing and distributing FLOSS implies some modifications of government procedures and taking on tasks that are new and for which often no organisational resources exist. It is, for instance, important to provide good business and technical support and to give training to the users. One result of the survey that was carried out for the purpose of this project must be highlighted here, too, because it sheds a light not only on the motivations of PAs for developing and distributing FLOSS but also on an important aspect of their self-perception: As described in the case studies analysis (D3), local governments have mainly two reasons to release software under a FLOSS license. They share the philosophy of FLOSS (“to contribute to FLOSS (freedom of choice regarding software)”), and they want to increase service quality (“to enhance the quality of the services provided by organisations like mine”). Thus, at least on the motivational level we find in PAs a strong linkage between FLOSS and service quality, i.e. FLOSS is not only seen as a technical alternative to existing software solutions within public administrations, but as integral element of service provision with a direct impact on service quality. FLOSS obviously is considered to provide opportunities on the organisational and procedural level of PAs.

Opportunities of the organisational change that springs from engaging in FLOSS production and distribution consist, for instance, in a more direct contact and increased interaction of PAs with other PAs, citizens, businesses, and the local and regional or national FLOSS community. When a PA successfully establishes new organisational structures that permit improved interaction, needs of citizens, businesses, and other PAs could faster and better be recognised, and it could become easier for PAs to meet the demands of these groups.

Collaboration and exchange of data between PAs might be eased through FLOSS projects and usage, which is often aligned with increased usage of open standards. Provided that PAs reorganise in the course of FLOSS projects in a way that secures a better response to user needs, this organisational change might go hand in hand with technical changes aiming at increased interoperability and thus make PAs more effective and eGovernment services more transparent.

PAs' involvement in FLOSS offers businesses and FLOSS developers new opportunities to collaborate with the public sector, to realise product ideas (i.e. increased innovation, as described in our theoretical model) while the public sector in turn finds itself in an improved situation with regard to procurement of software because the number of producers, especially SMEs, will increase (which was is also discussed in our theoretical model).

The indirect effects of PAs publishing their own software under a FLOSS licence on transparency and on the economy will be discussed in detail in chapter 7.

6.6.2 Barriers

All the new activities and responsibilities described above require from PAs changes in the organisation of tasks in the IT departments, as well as in the other departments whose services are subject to changes in course of the FLOSS project. People who were previously tasked with maintenance or performing internal processes become responsible for communication with actors outside the PA, for providing content for websites and mailing lists, and for adapting processes to new technologies and needs. The procedures which have to be put in place the support and the training are not different from the support that a company provides for a non FLOSS product, except that

- the documentation should be written in English, if the aim is to spread the product across different countries with different languages and to take advantage of an international FLOSS community
- The developers must check the websites regularly in order to quickly answer the questions of the users or potential users

Our case studies have shown that PAs have developed different solutions for these challenges. MMBase and APLAWS manage the content they provide on-line like technical support. Like CommunesPlone,they rely on third party companies to locally provide more specific support and training and to help distributing the product. The provincial government of Extremadura established an intermediary (FUNDECYT) in order to supervise and coordinate the project. NFI collaborates closely with the end users and develops GUIs (graphical user interfaces) and other features in order to improve usability of the software for laymen. Beaumont Hospital has created user manuals for training institutions, but compared to the other case studies it often lacks interaction with external actors, and – maybe as a consequence of this – the IT department complains about weak support from other users / the community, while take-up of their software is quite remarkable.

Developing and distributing FLOSS is aligned with new requirements to the management of PAs.

Experts consider that change management and user preparation and education are a cornerstone of successful FLOSS projects. Similarly it is equally important to assess the preparedness of public sector IT managers and decision makers to invest in specific FLOSS project development and management knowledge and to generate motivation and incentives.

Apart from these challenges, publishing the code to external organisations has a direct impact on procedures at the level of the public bodies that publish the software because

- they have to keep track on the modifications made on their products (the product can be forked, which has an impact on the ownership of the next releases of the product)
- they have to guarantee the quality of the product and
- they have to take into account the internal and external user needs.

A crucial factor in this regard is the level of experience PAs dispose of when they start their FLOSS development and distribution projects. Lack of prior experience with FLOSS is likely to be a barrier to using – and releasing – FLOSS software. The degree of familiarity with FLOSS at the time when the PAs we examined in our case studies started their FLOSS projects varied considerably, ranging from zero experience (like at Beaumont Hospital or the provincial government of Extremadura) to good expertise (like at NFI or London Borough of Camden). It must be emphasised that experience here means not only the experience in the technical aspects of FLOSS, i.e. programming languages, coding conventions, etc. All our case studies, except for Beaumont Hospital, were familiar with FLOSS, its ideas and principles. This familiarity was usually provided by individuals in the organisation who, in a certain context of decision-making on the implementation of services into software, were able to convince the organisation as a whole to opt for FLOSS.

7 Impact of public administrations making their own software available as FLOSS

In the following sections, we examine what impact the development and distribution of FLOSS by public bodies has on eGovernment services, the economy, and the information society.

7.1 Impact on eGovernment services

The impact of developing and publishing FLOSS on eGovernment services must be considered with regard to changes related to the software configuration and hardware architecture – and thus on the degree of FLOSS usage - on the one hand and changes in the service quality, especially with regard to interoperability and transparency, on the other hand.

The most fundamental change of software configuration and hardware architecture could be observed in Extremadura, where the number of PCs in the educational sector reached a very high level (e.g. 1 PC per 2 students in schools) that could not have been achieved in such a short time (or at all, within the limited budget) with proprietary software. Moreover, in this case, completely new IT networks and communication facilities (website) were set up. Beaumont Hospital also experienced a high degree of change in their software configuration and hardware architecture because the IT department replaced almost all proprietary software by FLOSS due to budget restrictions. A technical reason for these changes that became an issue later in the course of migrating towards FLOSS has been explained by the head of the IT department, who told us they found out that “most of the Microsoft operation systems we found couldn't handle two instances running on the same box. We were deploying lots of little servers and they were just growing. When we shifted towards Linux we found that we could multitask to a much greater degree.”

The initial aim of APLAWS and MMBase was to develop products, rather than internal applications. They built their IT systems accordingly: Building management systems, release management tools, staging servers, etc. APLAWS and MMBase release their software without changing hardware or software architecture. Instead, they used the SourceForge infrastructure in order to provide download and support facilities to their users.

The lowest degree of change of software configuration and hardware architecture has been observed at NFI. Since the software that NFI develops is application software for criminal investigations– by other government agencies – the development of FLOSS has no significant impact on the IT system used at NFI itself.

In conclusion, the degree to which software configuration and hardware architecture are affected by FLOSS development and distribution projects in the public sector depends on the scope and purpose of these projects. If the project aims at reorganising the services the public body provides, then the effect on software configuration and hardware architecture is quite high. In contrast, when FLOSS is produced for the purpose of serving the needs of other users / institutions or as an application that runs by and large independently from the services and processes of other departments, the impact on software configuration and hardware architecture is rather low.

The strongest impact of FLOSS activities on the scope of FLOSS usage in public sector organisations could be observed at the provincial government of Extremadura. In the beginning, the government didn't know what impact it would make by setting-up its own GNU/Linux distribution. Representatives interviewed for this study depicted very impressively how surprised the government was by the dynamics GNU/LinEX developed:“We started at the schools and we made a

distribution to the local newspapers of the Gnu/LinEx CD" but quickly the "...civil society and also enterprises wanted to work with [the Gnu/LinEx distribution]..."

In Extremadura led the GNU/LinEx project to substantial changes in the use of FLOSS within the government. Originally limited to education, FLOSS solutions were adopted for healthcare and hospital management, and in 2006 a decision was made to require the entire public administration to use FLOSS, including for ordinary government activities.

A second case that appears to be special is Beaumont Hospital, where the extent of FLOSS usage increased a lot (including GNU/Linux on the desktop) after the decision to migrate to FLOSS was made, but where repercussions in form of resistance to FLOSS could be observed after the individual users were forced to change what they were used to. This form of user resistance to change is often encountered when users are not sufficiently involved in the decision making and implementation process for change, and is not specifically related to FLOSS. Nevertheless, as a consequence of this user resistance, the hospital's management withdrew its support of the full FLOSS migration, and some FLOSS systems on the desktops had to be partially replaced by proprietary software again.

The other public bodies we interviewed (except for CommunesPlone, where PAs specifically aim to increase FLOSS usage) typically reported limited impact of the FLOSS projects on overall FLOSS usage. However, the IT experts in the public bodies denoted that they were generally interested in increasing the extent of FLOSS usage. Apparently, once the IT departments became familiar with FLOSS they discovered that it provides a lot of opportunities regarding cost, flexibility and maintainability. It should however be noted that this issue was often considered to be a bit delicate, as the IT experts were aware that they could increase the extent of FLOSS only if they kept in line with user demands and capacities. In some cases it turned out that it was hard to convince the management or higher levels of administration of the benefits of FLOSS. Especially when higher hierarchical levels came into play, "lobbying against FLOSS" (which has hardly been observed directly in the public bodies we examined) suddenly became an issue. This seems to reflect actual lobbying strategies of proprietary software companies, but it may also reflect a general distrust towards budget and purchasing decisions of higher administrative levels. It is noteworthy that our survey findings indicate that at the broader level, the use of FLOSS is associated with FLOSS development, in that FLOSS users are more likely to release FLOSS than those not already using FLOSS. But it is not clear whether there is any consistent causal relationship between usage and FLOSS development in the public sector.

Regarding the impact on eGovernment services, we have to recall that in the Dutch OSOSS surveys as well as in the EU-wide FLOSSPOLS local governments survey interoperability turned out to be one of the most important reasons for public administrations to consider FLOSS. Respondents who valued interoperability or said they were too dependent on their vendors were significantly more likely than others to plan to increase their use of FLOSS in the near future. Improved exchangeability of data between departments as well as between different organisations was considered to be achievable through the use of FLOSS. Although FLOSS and interoperability are not the same thing, they are clearly seen by public sector users as being related in that interoperability, open standards and the resulting vendor independence are perceived to be associated more with FLOSS than proprietary software. This may reflect the current reality that although open standards and interoperability are *technologically* fully compatible with proprietary software, they often do not appear to fit into the business strategies of proprietary software vendors. All case study interviewees also emphasised the importance of interoperability, although they did not see this issue related specifically to FLOSS. The most important aspect regarding

interoperability was clearly seen in using or not using open standards. Several European countries provide interoperability frameworks and push the usage of FLOSS, these do not discuss the release of software as FLOSS.

Moreover, there is a significant difference between PAs that engage in developing FLOSS and those that do not – the latter providing the clear majority of PAs. As described in the previous section, those PAs that engage in FLOSS feel that developing and using FLOSS and service quality are inter-related. Yet our survey showed that for the majority of those public sector organisations that do not engage in FLOSS the opposite is true: They see themselves as a service provider and do not relate this to the opportunities provided by software development and distribution. With regard to software, they define themselves as users and consumers rather than as suppliers, and they do not consider software as an integral component of their services and the way they provide their services. More than 40 comments we received from this group on our survey were along the lines of "we are a public service provider and not a software house!" Political initiatives aiming at increasing the attractiveness for public bodies to develop and distribute their own software should therefore address this self-perception, as well as the perception of the role of software for the provision of public services in the information society.

The degree to which the publication of software used for processing citizen and business data contributes to the objective of transparency in government depends on how much this software, when in use, increases the degree to which businesses and citizens obtain insights in and become able to comprehend decision-making in concrete government activities. In this regard, representatives of the public bodies we examined often stated that the citizen / user of governments services usually cannot see and does not care what kind of software works behind the interfaces they use to perform tasks. The same applies to the impact on transparency of public administration. Usually, PAs do not perceive this as an issue related specifically to FLOSS. However, where software is implementing specific aspects of the law, transparency is clearly increased by the source code of the software being available to the public for inspection. For instance, software used for running elections is an essential part of the voting process, and transparency of this process may require public access to the software source code (though this transparency does not require the further attributes of a FLOSS licence of allowing user modification and distribution)..

The impact of developing and using FLOSS in public bodies on the transparency of their services is thus probably limited to specific areas. In terms of public perception, for software in general, "usage of Open Source or [proprietary] software has no impact on the transparency of the government activities ... citizens are only concerned with the security of their transactions independently of any software considerations", said the representative of the provincial government of Extremadura, agreeing with other public bodies that were examined for the study.

Most PAs believe that interoperability relates to open standards and while this may be supported by (and should support) FLOSS, it is independent as an issue. Belgium, for example, increased the interoperability of their activities thanks to their "Banques carrefour" - literally "crossroads bank", large Enterprise Application Integration (EAI) systems used to exchange information about, for example, the social security, taxes, private and public companies, etc. Thanks to those services and the automatic electronic exchange of information between the administrations, citizens and companies have drastically reduced the time needed to obtain legal documents.[23] This, however, has no impact on the view of the citizens on the transparency of the government activities through the usage of open standards or FLOSS.

[23] The "Banques carrefour" use open standards defined by ISO, W3C or OASIS (Frank Roben 2006 page 29 - Spécifications ouvertes et standards ouverts)

The fact that department heads and other employees at PAs regard the effect of the technology they use on how the service provision and its quality is perceived by the users as being so low may be explained by a combination of cultural and organisational obstacles on the one hand, and technological barriers on the other hand. Many services are provided to the user/citizen in one direction only, i.e. from the PA to the user/citizen (see Capgemini 2004), which limits the degree of the user's/citizen's involvement in and information about the service transaction, but which also limits the awareness of the PA of the needs, preferences and capabilities of the user/citizen. We assume that as more "front-office services" become offered to the user/citizen in a bi-directional way and involve her or him directly in the service transaction, there will be more awareness by PAs for the interoperability and transparency of their service products and procedures, as European initiatives for eGovernment emphasise those characteristics (see below). In this respect, the potential for FLOSS to unfold in the public sector may depend on broader eGovernment initiatives and willingness to provide more and more government services by electronic means. On the cultural and organisational side, this would require a different understanding of the provision of public services, from PAs as well as from users/citizens.

Evidently, interoperability and transparency are usually considered to be related to a broad set of political and technological requirements and objectives that cannot be reached by "simply" developing and publishing FLOSS. As main measures for interoperability and transparency we consider the degree (intensity) of cooperation between different public administrations (PAs) and – where appropriate - the existence of public-private partnerships in provision of an eGovernment service. The latter is relevant because public-private partnerships require collaboration of public authorities with external persons or organisations. This aspect highlights that high requirements already arise if collaboration with businesses are being envisioned. If interoperability and transparency in interaction with the individual citizen become an issue, these requirements may even increase (van Oranje et al., 2006). IDABC's "European Interoperability Framework for Pan-European eGovernment Services" provides a good example of the multitude of challenges that must be mastered if interoperability and transparency are to be achieved, and indicate that migrating to and / or developing of FLOSS alone will probably not suffice in this respect. Some challenges relate to cultural/organisational aspects, while others highlight technical requirements.

On the level of cultural/organisational issues, IDABC suggests that service requirements should be jointly determined by the participating administrations via a demand-driven approach; that all involved partners should agree on Business Interoperability Interfaces (BII) that enable the business processes to interoperate across administrative units (which may require studying the definition of common BII standards); that there should be service level agreements in order to formalise mutual expectations from the service providers; and that the service providers should agree on a common security policy. However, the one-directional way of service provision (from PAs to citizens) that was observed by Capgemini and our observation that PAs often tend to focus on themselves and ignore other players around them indicate that as a precondition for meeting these requirements, many PAs will need to undergo a considerable cultural and organisational change.

The technical requirements IDABC specifies should be met in order to achieve interoperability might be easier to reach, but they suggest also that developing or adopting FLOSS in itself will probably not suffice in order to achieve significant gains of interoperability and transparency of government services:

- the data elements to be exchanged should be made interoperable
- the service providers should find a single language, based on XML, in order to make (legal) vocabularies interoperable

- technical interoperability should be provided at front-office level as well as at back-office level
- the technical interoperability of eGovernment services should be based on common guidelines
- these common guidelines should be based on recognised open standards

7.2 Impact on the economy

The survey that was carried out for the purpose of this study showed that releasers of FLOSS assume a positive economic impact when local governments release their own software under a FLOSS license. They think that local enterprises could then provide support, training etc., and create new employment. None of the "releasers" thinks that releasing FLOSS by public bodies would have a negative impact on the local economy because proprietary software vendors could reduce their turnover and cut employment. However, almost as many "non-releasers" (43%) assume that there would be no impact on employment at all if local governments release their own software as FLOSS. Only roughly one fifth of the "non-releasers" think releasing FLOSS through PAs would have a positive impact on the local economy. Yet, the share of those non-releasers who expect a negative impact is very low (3.1 per cent). What characterises the expectations of the "non-releasers" is that they assume there would be no economic impact at all (53 per cent) while roughly one quarter of them do not know what the economic impact of releasing FLOSS by public bodies would be.

The main conclusion that must be drawn from these findings is that neither the "releasers" nor the "non-releasers" expect a negative economic impact of public bodies releasing their own software as FLOSS, and that there is some confidence that it might have some positive economic impact, if any.

In order to examine the possible economic impact further, we classify applications domains by extent of specificity to the public sector (see Table 10), which addresses the reservation of our theoretical model that the economic impact of PAs making software available as FLOSS depends on the scope to which their FLOSS products can be applied.

Very generic purpose	e.g. operating system, desktop environment, Office suite
Specialised purpose	e.g. content management, a collaborative work environment, a work flow system that could be used by various public bodies but also by firms and other users
Specific PA purpose	e.g. land record management, public health

Table 6: Types of software applications

Source: Ghosh, Glott, Robles & Schmitz 2004

Public bodies' involvement in the first kind of activities would reinforce the economic dynamics in existing FLOSS development projects that are related to the existing standard software programs, the second kind of activities would initiate new development projects and activities and set up new economic dynamics in the software markets. Notably, the organisational challenges public organisations are confronted with differ also very much depending on which kind of software they develop and how they integrate in the FLOSS community.

The possible effect of increased economic growth and employment through the broad usage of

FLOSS by the public sector[24] may be reinforced as public organisations change their role from a pure "platform provider" to a "solutions provider". As "platform providers", public organisations rely on the willingness of other actors in the field of FLOSS to respond to their interests and to develop the software they need. As "solutions providers", public organisations become independent initiators of (in-house or sub-contracted) software development and may considerably increase the number of software product varieties. The shift from "platform providers" to "solutions providers" may require training courses for IT department staff and related educational infrastructure in the region where public organisations that perform this shift are located. The resulting amplifying effect of skills distributed and public participation in the information society may increase in future if public organisations increase the scope of their activities in the field of FLOSS. This effect, and its local, regional, national and international scope, has been examined in the study by focusing on spill-over effects that could be observed from the respective FLOSS projects of the PAs that were scrutinised.

Particularly in the case of the provincial government of Extremadura, we find benefits of FLOSS release for the public sector (after the education services have shown great success, health services are now migrating to FLOSS) and the local economy. The provincial government of Extremadura collaborates with a broad scope of external partners:

- ANDAGO, a software house employed in order to create the first release of GNU/LinEx
- the (Spanish) FLOSS community
- private companies commissioned by the government in order to provide proprietary software for the expansion of GNU / LinEx to the health sector, where privacy protection laws limit the usage of FLOSS and require to use proprietary software

As described in the case study analysis (D3), local businesses and the local FLOSS community benefit from the general need of PAs to adapt software to their specific needs, and these spill-over effects are very important at the political level of the project.

Significant spill-over effects were also observed at the APLAWS project of the London Borough of Camden, as APLAWS started to grow into the educational sector and was taken up by companies, such as Carrefour in France or the Deutsche Post in Germany. Another interviewee told us that these versions differ considerably from the version distributed on the website, so presumably these companies must have employed own or external developers to change the system. However, though the project coordinators conclude from the huge number of APLAWS downloads (several thousands) that there must be spill-over effects, they have no overview of what the users do with it and how changes are organised.

Another spill-over effect of APLAWS takes place in the consultancy industry and in the training sector, as the project coordinators know of freelancing consultants who offer advice and training for APLAWS. These freelancers seem to have a public sector background (or have at least specialised on clients from the public sector), downloaded the program and studied its workings in order to provide services for others who want to use it. As with the provincial government of Extremadura, external actors play a significant role for APLAWS, too.

The other case studies that we analysed did not have a good overview of spill-over effects of their FLOSS projects, though at least one case, the Beaumont Hospital, referred to take-up of their

[24] This view is supported by an empirical study on FLOSS usage in Italy (CENSIS, 2004). Weber (2004) supports this hypothesis while stressing a different aspect. He points out that the democratic principles of FLOSS have a strong potential in other industries than the IT industries, especially in biotechnology and publishing.

products in other hospitals and health insurance companies. In the case of NFI, there are probably no spill-over effects at all because the market for the products NFI offers is very small. Finally, CommunesPlone cannot be estimated with regard to spill-over effects because this project is designed to attract private sector companies to become part of the CommunesPlone network. Nevertheless, CommunesPlone provides thus an example how economic impact, even on an international scale, can be achieved in quite a short period of time when PAs strive to integrate businesses in their FLOSS strategy from the beginning .

Finally, one of the most important aspects to consider regarding the economic impact of PAs' FLOSS activities is of course their effect on employment. Interestingly, the awareness of the representatives of the PAs we interviewed of such effects was very low, and their assessment regarding employment effects was quite negative. Whenever we asked for employment effects they tended to deny such effects. However, most of them had a “public administration perspective” instead of a “project perspective” on this topic and considered jobs that were created in the course of the project not caused by this project, but by general demand for administrative personnel, especially when the hired persons were employed for other tasks than the software project, too, or when such personnel decisions were made at superior hierarchical levels or in other departments.

Overall, with regard to spill-over and employment effects the results of the analysis of the economic impact of public bodies making their own software available as FLOSS confirmed our assumption that the impact increases with growing scope of applicability of the produced software. In other words: Very generic and special purpose FLOSS products of PAs have a medium to strong impact on the economy. The finding that PA-specific software has no discernible impact on the economy. does of course not mean that PA-specific software is irrelevant. These niche products contribute considerably to a better supply of software that is needed by PAs, and often meet a demand of PAs that is not met by proprietary software producers because the market for these products is too small, so that profits could only be made when the price becomes extremely high – which many PAs cannot afford. In this regard, CommunesPlone provides an example of how PA networks (in combination with businesses and developers) can directly approach this problem.

We should add a caveat, however, that although both the case studies and the survey showed that public sector organisations releasing (or planning to release) software as FLOSS expect a positive impact on the economy, having such a positive impact was almost never reported as a reason to release software. It was one of the least important motivators in the survey, and among the case studies only Extremadura intended its FLOSS activities to result in benefits for the local economy. For the most, local economic growth is an expected, positive side effect of a release strategy that is decided for other reasons.

7.3 Impact on the information society

Though the case studies we examined showed a number of similarities, especially with regard to the challenges they had to master, they also show significant differences that play an important role when their actual or potential impact on the information society is to be assessed. Therefore we would like to elaborate the typical elements of each case that makes it distinct from the other cases. These distinct features cause different effects on shape and principles of the emerging information society.[25]

[25] We do not claim that the following typology is exhaustive. It elaborates on the discussion in the separate report presenting and analysing the case studies, and its main purpose is to serve as a starting point for the development of scenarios for the impact that public bodies may have on the Information Society when developing and releasing own software under an OS license .

As was explained by a representative of the provincial government of Extremadura, the cornerstone of GNU/LinEx is to adapt software, not to innovate on software. Adapting software is aligned with a very clear vision of shaping the information society in the province of Extremadura, i.e. with a spatial and societal purpose. The provincial government lightly customised the user interface of the products they used by changing the name of the basic application (replacing them, for example, with famous Spanish citizens, artists, etc.). In contrast to this, NFI, Camden, MMBase, and Beaumont Hospital develop brand new products which imply a huge technical impact depending on their user needs and their wishes to keep an eye on the external development based on their products (CommunesPlone is discussed further below). We would therefore like to typify GNU/LinEx as "externalised customisation" of an existing software product. Basic characteristics of this type are

- the public body has a need for software but no capacity to develop, distribute, and maintain it,
- therefore the development (of the first release) is completely contracted out,
- but requirements are clearly specified by the PA and the PA becomes "owner" of the software and plays a vital role in maintaining the project after the first release is published
- this type seems to be mainly driven by general political strategies, the software (OSS) itself appears as a means that helps to achieve more general political objectives

What characterises the APLAWS project of the London Borough of Camden, but also the MMBASE project of VPRO and the FLOSS development at NFI is that these PAs had at its disposal a team that was capable to design, implement, and carry out an innovative FLOSS development and distribution project without the need to create knowledge on FLOSS within the organisation or to hire FLOSS-experienced personnel. We'd like to call this type of readiness "experience-based innovation". This type is characterised by following features:

- the public body has a need for software and the capacity to develop, distribute, and maintain it,
- the PA holds a position as coordinator over the whole project period and tries to ensure that erroneous developments or forking do not occur
- this type seems to be mainly driven by a demand for (a) particular functionality (functionalities), the software (FLOSS) is not means for other objectives but objective itself

A third type of programming and maintenance capacities for public sector FLOSS activities is provided by Beaumont Hospital. Beaumont Hospital had no FLOSS experience at all before they decided to migrate to FLOSS. They actually had a Linux server, but nobody was aware that it was FLOSS and how the server worked. The reason to migrate towards FLOSS was a severe cost pressure, and only after the IT department had had good experiences with FLOSS after this decision they began to extend their FLOSS activities and to promote their products. Therefore, the development of software was faced with massive organisational and technical barriers, because nobody in the IT department had any knowledge of FLOSS, no FLOSS systems were used by 2001, the staff had very limited systems management and systems engineering skills, and many people only understood GUIs. While at the beginning of the FLOSS activities migration to FLOSS dominated, the team subsequently began to develop own and innovative software and to provide even training tools in order to enhance the spread and usage of their products. We'd like to call this type "cold adoption", which is characterised by

- the public body has a need for software but neither capacities (or the willingness) to develop

this software in-house nor the financial means to meet the demand with proprietary software

- therefore the inexperienced IT staff has to attain programming and software engineering skills within a short period of time
- this type seems to be driven by scarcity of resources (either financial or organisational), FLOSS is chosen in order to secure the "survival" of the organisation (or department) and / or the services that are provided within and by the PA

The software development and maintenance capacities of the municipalities that form the core of the CommunesPlone network are limited. This lack has been balanced through a close interaction of these municipalities with the PLONE FLOSS community, Zea Partners (formerly Zope Europe) an international network of SMEs. From the start, the whole project was modelled on the PLONE community organization and development practices. We like to consider this as a fourth type of programming and maintenance capacities for public sector FLOSS activities, which appears to be something of a mixture between Extremadura's "externalised customisation" and Beaumont Hospital's "cold adoption". In this model it is comparably unimportant whether or not a public sector organisation has software development and maintenance experience, because these tasks are more or less completely externalised. Another aspect in which this model resembles the Extremadura case is that it has a spatial – or in other words – societal – dimension right from the beginning, as it aims at integrating more and more municipalities and commercial partners across Europe. Like in the GNU/LinEx case, CommunesPlone bears the potential to save cost and to improve the quality of service provision. Since it primarily aims to gain independence from software vendors by means of providing a growing pool of software that can be shared, we would like to call this model "externalised software-pooling". This type is characterised by following features:

- the public body has a need for software that can be shared with others at low cost in order to become independent from software vendors
- the IT staff and the management has to define which software should be developed and shared, but the public agency does not need to have experience in software development, maintenance, and release
- this type is driven by economic factors, the wish to save cost through sharing of software that can be purchased at low cost, but due to the importance of sharing this model aims "by nature" at a large spatial dimension

As previously stated in this report, the impact of PAs making their own software available as FLOSS appears to be limited by the scope of the usability of the developed software. With regard to the spatial and societal objectives pursued by gnuLinEx and CommunesPlone we can extend this rule and add that the possible impact is also limited by the scope of the infrastructural change that it is intended to achieve. Our observation is that most of the desired functionalities that initiate FLOSS development activities come into existence in the context of sub-tasks that are performed within PAs. It is not to be expected that FLOSS activities of PAs *per se* reach a societal scope. Rather, the opposite appears likely. The survey that was carried out for this study revealed that the local economy (i.e. the "first step" outside the boundaries of a PA, still far from the realm of "Information Society") is not what PAs have in mind when they think about software development projects. We interpret this as an indication of the limited societal impact of public bodies' FLOSS initiatives.

Nevertheless, it is also obvious that the public sector has only just started to become an active FLOSS developer, and there is an opportunity for such initiatives to spread and to be performed by

ever more PAs if there is a need for special software functionalities. Though we consider that *currently* the *actual* impact of such attempts as limited, we see a great *potential* for a fundamental impact of public bodies' FLOSS activities *in future*. Each of the four types of FLOSS development and distribution by PAs has a distinct potential impact on the Information Society:

A) "externalised customisation": Only the provincial government of Extremadura set objectives with its FLOSS strategy that go far beyond the scope of the demand for special functionalities, as the LinEx project was integrated into a general "Information Society Strategy". As is well-known, the project resulted in a major improvement of the IT infrastructure in the public sector, especially the educational sector, and in improved computer literacy of the population. It is this "embeddedness" of public FLOSS development activities in a broader political strategy what characterizes this approach and what enables it to achieve significant impact on the Information Society, though on a regional scope only. The political element overcomes the limitation of the special need that the software is required to meet and transfers the impact of FLOSS from the level of its direct application to the social context in which it is applied. It is noteworthy that the limitation on the region in the case of LinEx does not derive from the FLOSS development project; it is set by the political strategy. As a consequence, if such development strategies are embedded in political strategies that target the Information Society on national, multinational or even global scope, it can be expected that FLOSS turns out to have a similar impact on a larger scale.

B) "experience-based innovation": The potential impact of FLOSS in this scenario is much more tightly bound to the underlying special functionality that initiates PAs' FLOSS projects. A significant impact of public FLOSS projects that correspond to this type is only likely if the install-base of the developed software exceeds a critical level and becomes more or less a standard application. Probably, in order to have a "societal" effect, a certain degree of community involvement must also be reached. APLAWS as well as MMBASE appear to have the potential to become such "standard applications", but they also demonstrate how difficult it is to involve and affect actors outside the closed user community in the PAs and other organisations that develop and / or apply it. NFI software like TULP2G, which also belongs to this type of public software development projects, is even more limited in this respect than APLAWS or MMBASE, because this software mainly serves the demand of small niches in the software market.

C) The scenario based on the "cold adoption"-type is flawed by the fact that the starting conditions of FLOSS development projects of this type are quite poor. This type is characterised by the fact that the staff has to catch up and attain new skills before any code can be written. After this process is completed, migration of existing systems rather than developing new software becomes an issue. However, once the migration process is completed the development process would be converted into the "experience-based innovation"-type, with all its limitations and potentials.

D) The scenario based on the "externalised pooling"-type is not so much determined by the software and its functionalities (in principle any kind of software that is needed in public authorities could be developed and shared) but by the spatial scope that is succeeds to cover. Thus, the growth of the user community and of the network is critical for the societal impact of such models. Theoretically, all public sector agencies in all countries could decide to participate in and benefit from this sort of FLOSS development and distribution. This type is characterised by the fact that the staff in the PAs is required to specify which software to

produce and share, but it is not required to organise production, distribution, and maintenance itself.

Finally, we should emphasise that the greatest impact on information society clearly occurs when the decision to release software as FLOSS has not been made for purely technical or practical reasons, but with wider impact on information society built in to the strategy from the start. As seen in the case of Extremadura, this may require more general purpose software as well as a greater commitment to engage with different parts of the information society, from a community of developers and businesses to citizens and policy makers. Impact on information society must, therefore, be designed into the process of software release, and is not as likely to happen by accident.

8 Conclusions

At the start of this report, we considered a number of questions that we intended to explore during the course of this study; questions that were raised by the prospect of public sector organisations releasing their software under FLOSS licences. These concerned legal issues; practical issues of technology, deployment, skills and resources; and the impact software release may have on the economy and information society.

Regarding the three main aspects - legal issues, impact, and sustainability – we first have to conclude that despite the fact that developing and distributing FLOSS is associated with a number of potential difficulties for public sector organisations – of which they are often not aware – legal issues did not prove to be a major barrier for them to engage in FLOSS development projects. The present legal environment appears sufficient to enable public bodies to release their software as FLOSS. Instead, we identified as the most important barrier a lack of IT professionals that are experienced in software development and / or FLOSS.

Regarding the impact on the economy and on the information society, we found that the kind of the software that is developed and distributed (very generic, special application, specific public sector application) determines the degree to which a societal and economic impact can be achieved. The greater the applicability of a software product, the higher is the impact that can be expected. Except for those development and distribution projects (LinEx and, to a lesser extent, CommunesPlone) that were designed from scratch with a clear political, societal, and spatial purpose to change the infrastructure of the public sector (or at least of parts of it), the actual impact on the information society is rather low; but we see a great potential for such initiatives in future if policy-makers and managers at PAs become more aware of the specific needs of different types of such projects. The EU and its member states can help to provide additional dissemination by supporting FLOSS competence centres.

The use and development of FLOSS in public bodies should not be discouraged; instead, it is important to build awareness for the advantages that this software model may bring. Governments (on all levels, i.e. local, regional, national, EU) should make decision-makers in public administrations aware of the advantages of developing and distributing their own software under a FLOSS license. These potential advantages in particular concern service, design, processes, quality, interoperability, cost, efficiency and vendor-independence. PAs can have an impact on local economies through developing and releasing own FLOSS, providing opportunities for firms, improved economic and educational chances for FLOSS community members, employment opportunities.

Overall, managers and employees in PAs still tend to focus on their organisation and to disregard the potential of direct interaction with citizens and users and the FLOSS community. Furthermore, if the release of software is part of a strategy to achieve a wider economic and societal impact, then it is more likely to do so than if it is conducted for purely technical or practical reasons. Putting it differently: If no impact on the information society is intended, chances are that none will result. It is not an automatic side-effect of releasing public sector software as FLOSS.

The availability of skills is the bottleneck for the public sector to become able to develop and distribute FLOSS. Except for LinEx, all FLOSS projects of public bodies we examined in case studies were initiated by the IT departments of these organisations. This observation, which is in line with previous findings from other surveys, suggests that IT, or more specifically, software expertise is a very important precondition for public bodies to move towards releasing software.

While this may seem obvious, it is not necessarily widely understood. Moreover, this expertise does not have to be FLOSS expertise. As the examples of LinEx and Beaumont Hospital indicate, FLOSS projects can be quite successful even if no knowledge of FLOSS is available within the organisation when the project starts, as long as IT expertise is present.

Though skills are the most critical success factor for FLOSS projects in PAs, it appears that FLOSS also provides a solution for this problem: All representatives of the case studies reported an improvement of the skills base at their departments through their involvement in FLOSS project, especially with regard to software engineering skills. FLOSS apparently provides a learning environment that enables public bodies' IT departments to learn within a reasonable period of time how FLOSS works, how it can be developed and help to find solutions for the special functionalities that are needed, and how it can be distributed and maintained. (Indeed, other studies[26] have shown that FLOSS communities are effective environments for the informal learning of skills valued by employers, often more efficient than formal courses.) Thus, starting to deal directly with FLOSS would be a good way to overcome skills barriers that may hinder FLOSS projects in many public sector organisation.

Regarding sustainability, we found that all the case studies we have analysed showed state-of-the-art forms of software development. All organisations studied were able to build up and coordinate the resources required for developing and distributing FLOSS, including building networks with companies and developers. However, the user/developer communities we found in the public sector were not comparable to the large volunteers-based networks we know from big FLOSS projects such as the Linux kernel or Apache. Instead, the FLOSS communities of public sector software typically consist of employees of public sector organisations or their private sector suppliers and related businesses. Though FLOSS licences may make cooperation between different public bodies easier, even across national borders, this does not mean that the problems which are typically associated with such cooperation, and with software development in general, will not occur.

Of course, the more general-purpose the software, the larger the community of support and development, as in the case of Extremadura and LinEx, or even CommunesPlone which relies on the wider Zope/Plone developer community of individuals and businesses. It is clear that the formation of a community, as well as the successful sharing of development costs depend on the thought and effort invested into those aspects. They do not occur simply because software is published under a FLOSS licence.

The study identified the following factors as main motivators for public bodies to develop and distribute their own software as FLOSS:

- contributing to the FLOSS community (sharing the philosophy of OSS)
- enhancing service quality
- vendor-lock ins (i.e. the wish to become independent from software vendors)
- need for technical solutions that are not provided by proprietary software
- cost savings (savings of license fees, keeping old hardware for longer, synergy effects through reduced training efforts, etc.)

The fact that the idea of FLOSS was valued more than service quality should not be overestimated, since a considerable share of the respondents to our survey who were driven by the philosophy of FLOSS showed also a strong interest in enhanced service quality. Nevertheless, the fact that the philosophy of FLOSS appears to be a dominant driver may shed a light on the background of the

[26] E.g. the FLOSSPOLS Skills Survey, Ghosh & Glott 2005c

persons who are involved (or who initiate) such projects in public sector organisations, as one might assume from this that such persons are probably somehow related to the FLOSS community. However, our case studies showed that in most cases, initiators of FLOSS projects grew aware of the benefits and principles of FLOSS only after engaging with the community for a specific project. They had a much vaguer understanding of FLOSS at first, starting their development activities more often for reasons related to cost, functionality or sustainability.

Public sector organisations are often driven to release FLOSS for reasons similar to their reasons for *using* FLOSS, as typically all public sector organisations are primarily *users* of software, with software development playing a minor role. We found many hints that public sector organisations' self-perception is determined by their tasks as a provider of government services and that software is not considered to be an integral element of these services. It is evident from a theoretical point of view that public bodies publishing their own software as FLOSS contribute to increased competition and thus advanced market functionality, and maybe improved innovativeness as described in the theoretical model for producing FLOSS. But what we found in the case studies as well as in the survey is that obviously the aspects of the user-model play a more important role for explaining why public bodies decide to develop and distribute FLOSS. Except for Beaumont Hospital, which had to react to severe cost-cutting in its IT budget, each of the projects we examined had as its starting point a need for a functionality that was not provided by existing software or IT infrastructure. This observation complies perfectly with the theoretical considerations of increased usability through increased numbers of “vendors” of a software product, as described in the theoretical model for FLOSS usage.

Regarding the incidence of public sector organisations making their software available as FLOSS, we found that

1. the share of local governments that *are aware* that they own software that can be released under a FLOSS is low, but still significant (approximately 10%)
2. the share of local governments that own such software and has full rights on it is probably higher than the share of those local governments owning such software without having full rights on it, regardless of whether they developed the software themselves or contracted the development out. However, awareness of the existence of such software may be low, or such software may often be scripts or components that, while potentially useful to others, were not considered by our respondents as “software that can be released”.
3. there is apparently a potential for future software releases under a FLOSS licence among the vast majority of governments that currently do not own such software, as they acquire further software.

The type of software predominantly developed by or for public bodies is neither general-purpose nor public sector specific software, but specialised software such as content management or work-flow systems, heavily customised to be of use to the public sector but possibly also useful to other organisations.

Despite the fact that saving costs is a strong driving force for public bodies to migrate towards FLOSS, hardly any public body we examined was aware of the exact costs and benefits of its FLOSS activities. This appears to be because some of these costs or benefits are hard to calculate, and often the FLOSS project appeared as “necessary anyway”, so that its results were needed more or less regardless of the costs of some of its parts. Thus, the cost incurred in development was an obligatory cost, and the decision to release it as FLOSS did not add a further cost. Overall the cost balance that is estimated in the case studies is very positive in all cases.

There seem to be spill-over effects to local training and consultation agencies and to the FLOSS community, but due to the fact that much specialised software is produced such effects are not very prominent. The same applies to direct employment effects of the projects we examined. We think that for now public sector FLOSS activities are only beginning to have an impact on local economies, but that significant impact on businesses and employment could be reached if there was a broader take-up of FLOSS software activity, in the public sector, including software release.

Developing FLOSS (or releasing software once developed as FLOSS) is a challenge for any IT department in the public sector because it requires the organisation to (gradually) become familiar with its principles and standards. This is different from from just buying and using (proprietary) software – but is not necessarily different from owning software, developed in-house or by contractors, and trying to share it with peer organisations using any other, non-FLOSS form of licensing. Any form of sharing software requires some awareness of the issues involved and the potential responsibility – whether in terms of legal liability or simply new users' expectation of support from the organisation releasing the software. Public bodies that have no in-house programming capacities, in particular, have to find solutions to get the software they need developed, although this relates to their need to own software, not their decision to release it as FLOSS.

A major point of concern when public bodies develop their own FLOSS is to secure good technical documentation. Again, complexity and time frame play a role here, but also the degree to which a project can be planned in advance and to which work can be distributed. As APLAWS shows, even a very complex FLOSS project can handle documentation requirements if this task is attended to right from the beginning, and if there are partners with the experience to take over this responsibility.

As a matter of fact, none of our case studies showed any indication that technical problems could have been so strong that they could have caused the project to fail. On the contrary, we often found state-of-the-art development models as well as state-of-the-art FLOSS products in our case studies. Technical problems may decrease if in future more in-house software development capacities are built up in European public sectors. Alternatively, if more public sector bodies release software as FLOSS, there will be wider use of such software and a resulting increase in the need for customisation and support for such software, which could support the further development of software capacities in a supporting industry in local economies. This process may happen automatically, in line with software release, but is likely to happen faster if public sector organisations release software with an economic and societal impact built into their strategic policies.

References

Bayrischer Oberster Rechnungshof (2001): Jahresbericht 2001. München.

BearingPoint (2004):Server operating system Licensing & Support Cost Comparison. Windows Server 2003, Red Hat Enterprise Linux 3 and Novell/SUSE Linux 8. May 2004. Available online at: http://download.microsoft.com/download/7/3/9/739c7ab3-25c4-4b8c-9680-81ae10573b9d/BearingPoint.doc

Capgemini (2004): Online availability of public services: How is Europe progressing? Web based survey on electronic public services. Report of the fifth measurement. October 2004. Brussels: EC

Cowan, Robin. 1991. "Technological Competition and Variety: Issues of Diffusion and Intervention" in G. Bell, ed. Technology and Productivity: The Challenge for Policy, Paris: Organization for Economic Cooperation and Development, pp. 509-522.

Cowan, Robin. 1992. "High Technology and the Economics of Standardization" in M. Dierkes and U. Hoffman, eds. New Technology at the Outset: Social Forces in the Shaping of Technological Innovation Frankfurt/Main: Campus Verlag, pp. 279-300.

Cowan, Robin. 1995. "The Informatization of Government: From Choice of Technology to Economic Opportunity" Commissioned for the OECD/Tübitak Conference on "The Changing Role of Governments in Introducing New Information Technologies", Ankara, September 1993; in Science,Technology, and Industry Review, No. 16, pp. 195-223.

Cowan, Robin, 2002. "On the Number of Firms and the Amount of R&D" Economics Bulletin vol 12(6) pp. 1-9.

Cowan, Robin, Ferné, Georges and Foray, Dominique. 1991. Information Technology Standards: The Economic Dimension. Paris: Organization for Economic Cooperation and Development, [ICCP Report 25].

Cowan, Robin and van de Paal, Geert, 2000. European Innovation Policy in the Knowledge-Based Economy. Brussels: European Commission Directorate General: DG-Enterprise.

Cybersource (2002): Linux vs. Windows total cost of ownership comparison. Available online at: http://www.cyber.com.au/cyber/about/linux_vs_windows_pricing_comparison.pdf

Deutscher Bundestag (2002): Deutschlands Wirtschaft in der Informationsgesellschaft. Berlin, February 2, 2001: Deutscher Bundestag. Available online at: http://dip.bundestag.de/btd/14/052/1405246.pdf

Dusollier, S., Laurent, P., and Schmitz, P-E. 2004. Open Source Licensing of software developed by The European Commission (applied to the CIRCA solution). European Commission DG ENTR. Available online at http://europa.eu.int/idabc/servlets/Doc?id=19296

eGovernment News (11 May 2004): Cost of open source software for government debated in Ireland. Available online at:

http://europa.eu.int/ISPO/ida/jsps/index.jsp?fuseAction=showDocument-&documentID=2536&parent=chapter&preChapterID=0-140-194-349-360

Ettrich, Matthias (2004): Koordination und Kommunikation in Open Source Projekten. In: Gehring, Robert A. & Lutterbeck, Bernd (2004): Open Source Jahrbuch 2004. Zwischen Softwareentwicklung und Gesellschaftsmodell. Available online at: http://www.opensourcejahrbuch.de/2004/OpenSourceJahrbuch2004.pdf

Evans, David S. and Reddy, Bernard J. 2003. "Government Preferences for Promoting Open-Source Software: A Solution in Search of a Problem".
9 Mich. Telecomm. Tech. L. Rev. 313, available at http://www.mttlr.org/volnine/evans.pdf

Frank, Roben (2005): E-government dans le secteur social: développements futurs, 26 October 2005. available on-line at: http://www.ksz-bcss.fgov.be/documentation/ fr/documentation/Presse/ 60_ans_ONSS.ppt

Ghosh, R. A., (2003), "Copyleft and dual licensing for publicly funded software development", working paper, Available online at http://www.flossproject.org/papers/dual.htm

Ghosh, Rishab & Glott, Rüdiger (2003): Open standards and open source software in the Netherlands. A quantitative study on attitudes and usage in Dutch authorities on behalf of the office of the "Programma OSOSS", the Ministry of the Interior and Kingdom Relations, and the Ministry of Economic Affairs. November 24, 2003. Maastricht: MERIT. Online: http://www.ososs.nl/article.jsp?article=8804

Ghosh, Rishab & Glott, Rüdiger (2005a): Open Standaarden en Open Source Software in Nederland. Een kwantitatief onderzoek naar houding en gedrag van de Nederlandse overheid in 2004 in opdracht van het programma OSOSS, het Ministerie van Binnenlandse Zaken en Koninkrijksrelaties en het Ministerie van Economische Zaken. Den Haag: Stichting ICTU / Programma OSOSS. Available online at: http://www.ososs.nl/article.jsp?article=17363

Ghosh, Rishab & Glott, Rüdiger (2005b): Free / Libre and Open Source Software – Policy Support (FLOSSPOLS): Results and policy paper from survey of government authorities. Maastricht: MERIT, University of Maastricht.

Ghosh, R. A. and Glott, R., 2005c. "The open source community as an environment for skills development and employment generation", Proceedings of the European Academy of Management Annual Conference, May 4-7, Munich. Available at: www.euram2005.de; another version available as a deliverable of the FLOSSPOLS project at www.flosspols.org/deliverables.php

Ghosh, R.A., Glott, R, Robles-Martinez, G. & Schmitz, P-E 2004. Guideline For Public Administrations On Partnering With Free Software Developers. European Commission DG ENTR. Available online at http://europa.eu.int/idabc/servlets/Doc?id=19295

Ghosh, R. A.., et al. 2006: Study on the Economic impact of open source software on innovation and the competitiveness of the Information and Communication Technologies (ICT) sector in the EU - Final report. Available at: http://ec.europa.eu/enterprise/ict/policy/doc/2006-11-20-flossimpact.pdf

Hermans, T., 2002. “Measurement of GFCF in software in the Belgian National Accounts”. Joint ECE/Eurostat/OECD Meeting on National Accounts paper. Available at: http://www.unece.org/stats/documents/ces/ac.68/2002/12.e.pdf

IDABC, 2005. Proceedings of workshop, IDABC@LinuxTag: Working with the Community: OSS in Public Administrations. June. Available at http://europa.eu.int/idabc/servlets/Doc?id=21209

Koordinierungs- und Beratungsstelle (KBSt) der Bundesregierung für Informationstechnik in der Bundesverwaltung im Bundesministerium des Innern (2003): Migrationsleitfaden. Leitfaden für die Migration der Basissoftwarekomponenten auf Server- und Arbeitsplatz-Systemen. Version 1.0. – Juli 2003. Berlin: Bundesministerium des Innern. English version is available online at: http://www.bmi.bund.de/Annex/de_25072/Download_englisch.pdf

Mangham, A. and Ghosh, R.A. 2005. London Borough of Camden: public procurement of open source software development. Published on EC DG ENTR/IDABC Open Source Observatory, January 25. Available at http://europa.eu.int/idabc/en/document/3804/470

Nagler, M. 2005. Open Source Adoption of the German Federal Office for Information Security. Published on EC DG ENTR/IDABC Open Source Observatory, July 22. Available at http://europa.eu.int/idabc/en/chapter/470

Oranje, C. van; Simmons, S.; Cave, J.; Weehuizen, R.; Glott, R.; Rothenberg, J.; Rubin, J. (2006): EUReGOV: Innovative and adaptive pan-European services for the citizens in 2010 and beyond. Inception Report. Prepared for the eGovernment Unit, DG Information Society and Media, European Commission. RAND Europe and MERIT. Available online at: http://www.euregov.eu/deliverables/reports/inception_report_v2.pdf

Parker, R. & Grimm, B. 2000, “Recognition of Business and Government Expenditures for Software as Investment: Methodology and Quantitative Impacts, 1959-98”, US Bureau of Economic Analysis paper available at http://www.bea.doc.gov/bea/about/software.pdf

Parker, R., Grimm, B. & Wasshausen, D. 2002. “Information Processing Equipment and Software in the National Accounts”. National Bureau of Economic Research conference paper, available at: http://www.nber.org/~confer/2002/criws02/wasshausen.pdf

Robert Francis Group (2002): Total cost of ownership for Linux web servers in the enterprise. Available online at: http://www-1.ibm.com/linux/RFG-LinuxTCO-vFINAL-Jul2002.pdf

US Department of Commerce, 2000. U.S. Computer Software Industry: Size, Firms, Establishments, Employment, Receipts. Available at: http://exportit.ita.doc.gov/ocbe/USIndust.nsf/806cbc35babba9838525695100784a38/538b5d24b61020898525696 2006c91c8!OpenDocument

Varian, Hal R. & Shapiro, Carl (2003): Linux Adoption in the Public Sector: An Economic Analysis. 01 December 2003. Berkeley. Available online at: http://www.sims.berkeley.edu/~hal/Papers/2004/linuxadoption-in-the-public-sector.pdf

Weber, Steven (2004):The Success of Open Source. Harvard University Press.

Wheeler, David A (2002): Why Open Source Software / Free Software (OSS/FS)? Look at the numbers! Available online at: http://www.dwheeler.com/oss_fs_why.html

Wichmann, Thorsten (2003): Use of Open Source Software in Firms and Public Institutions. Evidence from Germany, Sweden and UK. FLOSS final report – part 1. Available online at: http://www.infonomics.nl/FLOSS/report/

Winslow, Maria (2004): Evaluating the ROI of Open Source on the Desktop. Linux and OpenOffice are ready for your business. In: Linuxworld.com, February 18, 2004. Available online at: http://www.linuxworld.com/story/43720.htm?DE=